Universidade do Minho
Escola de Engenharia

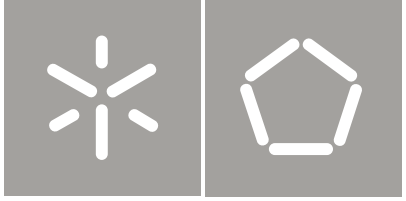
Luís Miguel Matos de Almeida

Célula Automatizada Para Montagem De Quadros Brancos Com Perfil De Alumínio

novembro de 2019

Célula Automatizada Para Montagem De Quadros Brancos
Com Perfil De Alumínio

UMinho | 2019



Universidade do Minho
Escola de Engenharia

Luís Miguel Matos de Almeida

Célula Automatizada Para Montagem De Quadros Brancos Com Perfil de Alumínio

Dissertação de Mestrado
Ciclo de Estudos Integrados Conducentes ao Grau de
Mestre em Engenharia Eletrónica Industrial e Computadores

Trabalho efectuado sob a orientação de
Professor Doutor Fernando Ribeiro

novembro de 2019

Direitos de Autor e Condições de Utilização do Trabalho por Terceiros

Agradecimentos

Quando nos propomos a realizar um trabalho deste género, contamos com o apoio e o estímulo de várias pessoas e entidades. É neste sentido que gostaria de agradecer a todos aqueles que contribuíram para a realização desta dissertação de mestrado.

Agradeço ao Professor Doutor Fernando Ribeiro, meu orientador na Universidade do Minho, pela ajuda e dedicação prestadas, ao longo da realização desta dissertação.

Não posso deixar de agradecer, também, ao Engenheiro Mário Martins, meu orientador na empresa *RobotSol*, pela orientação e transmissão de conhecimento, que foram essenciais no decorrer desta dissertação.

Os meus agradecimentos vão ainda para a empresa *RobotSol* e a toda a sua equipa, pela oportunidade fornecida e ajuda prestada.

Em especial, agradeço a três colegas que realizaram as suas dissertações na empresa *RobotSol*, Bernardo Pedrosa, Miguel Mira e Vitor Sousa quer pelo espírito de intreajuda e cooperação, quer ao longo deste período, ajudaram à realização da dissertação, quer pelos bons momentos partilhados.

Agradeço ainda, de forma muito especial, à Inês Cunha Ferreira, por estar ao meu lado nos momentos mais complicados e pela paciência e ajuda prestadas na elaboração da dissertação.

Agradeço, também, a todas as pessoas que me acompanharam durante o meu trajeto escolar. Ao longo destes anos ganhei não só colegas, mas também amigos para a vida.

Por último, mas não menos importante, agradeço à minha família, aos meus pais e ao meu irmão, pelo apoio, ajuda e incentivo prestados ao longo de todo este percurso.

Gostaria de dedicar esta dissertação à minha avó, Maria Rosa, que sempre acreditou em mim, e à minha mãe, Elisabete Almeida.

Declaração de Integridade

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Resumo

Desde sempre, o ser humano procura alternativas para substituir o trabalho que executa. Devido ao avanço tecnológico nos últimos anos e à tendência exigida pela globalização da produção ser contínua, a indústria tem vindo a apostar na automatização das suas fábricas. Com níveis de exigência elevados e com um maior número de processos, que eram antigamente realizados por mão-de-obra humana, a produção é cada vez maior, rápida, programada e contínua.

Dos vários processos que têm vindo a ser desenvolvidos de forma automatizada, encontram-se as operações de manipulação. Estas operações, envolvem a movimentação de objetos de um lugar para o outro, sendo *pick-and-place* um exemplo disso. Devido à exigência atual, os equipamentos têm de ser mais flexíveis, mais rápidos e mais precisos. Assim, a robótica assumiu-se como sendo a solução, pois é conhecida pelas suas características de precisão e repetibilidade de movimento.

A solução encontrada propõe retificar um sistema robotizado de montagem automático para substituição de linhas manuais tradicionais, em concreto, a montagem de quadros brancos com peças de plástico (topo e base) e perfis de alumínio (laterais). Estas peças, que compõem a estrutura externa dos quadros, são montadas e aparafusadas manualmente. Por ser um processo repetitivo, aborrecido e com uma produtividade baixa, é natural a necessidade de automatizar esta linha. Assim, foi prevista uma solução utilizando dois robôs (braço robótico - *KUKA KR 16-2*) e duas mesas para montagem e aparafusamento dos quadros.

Na solução geral, o primeiro robô agarra num quadro branco e o mesmo é colocado na primeira mesa, onde ocorre a montagem dos perfis de alumínio e das peças de plástico que compõem a estrutura externa do quadro. De seguida, o segundo robô coloca o quadro completo na segunda mesa, onde ocorre o aparafusamento do quadro e onde se amolga os perfis de alumínio, completando assim o quadro.

Esta dissertação descreve parte da solução prevista, que envolve um robô e a mesa de aparafusamento. Assim, é necessário a programação do robô, isto é, a marcação dos pontos de trajetória que é percorrida pelo mesmo, do controlo da mesa de aparafusamento, dos sinais do robô e do *HMI*, através do controlador lógico programável (*PLC*). Por fim, é feita a otimização da célula utilizando o programa *kuka sim 2.2*.

Palavras-Chave: KUKA, programação, simulação, PLC, automatização, pick-and-place, HMI.

Abstract

Human beings have always been searching for alternatives to replace the work they perform. Due to technological advances in recent years and the trend demanded by globalization for production to be continuous, the industry has been focusing on the automatization of factories. With high levels of demand and with a greater number of processes that were formerly performed by human labor, production is faster, programed and continuous.

Of the several processes that have been developed in an automated manner, there are the manipulation operations. These operations include moving objects from one place to another, being pick-and-place an example of this. Due to the current requirements, equipment has to be more flexible, faster and more accurate. Thus, robotics was assumed to be the solution as it is known for its precision and repeatability characteristics of motion.

The solution proposes to rectify a robotic automatic assembly system to replace traditional manual lines, specifically the assembly of whiteboards with plastic parts (top and bottom) and aluminum profiles (sides). These parts, which make up the external structure of the frames, are assembled and screwed by hand. As it is a repetitive, boring and with low productivity process, the need to automate this line is natural. Thus, a solution was foreseen using two robots (robotic arm - KUKA (KR 16-2)) and two tables for assembly and screwing the frames.

In the general solution, the first robot picks up a whiteboard and places it on the first table, where the aluminum profiles and plastic parts that make up the outer frame of the board are assembled. Then, the second robot places the complete frame on the second table, where the frame is screwed in and the aluminum profiles are dented, thus completing the frame.

This dissertation describes part of the intended solution, which involves a robot and the screwing table. This required the programming of the robot, including, the marking of the path points to be crossed by the robot, the control of the screwdriver, the robot signals and the HMI, through the Programmable Logic Controller (PLC). Finally, cell optimization was carried out using the program (kuka sim 2.2).

keywords: KUKA, programming, simulation, PLC, automatization, pick-and-place, HMI.

Conteúdo

1 Introdução **1**

1.1 Contextualização 1

1.2 Proposta de Trabalho e Descrição das Tarefas 3

1.3 Organização da dissertação 4

2 Estado da Arte **5**

2.1 Casos de estudo 5

2.1.1 1° caso de estudo 5

2.1.2 2° caso de estudo 6

2.1.3 3° caso de estudo 6

2.2 História da robótica 6

2.3 Robôs industriais 8

2.3.1 Componentes de um robô industrial 9

2.3.2 Juntas 11

2.3.3 Desempenho do robô 12

2.3.3.1 Repetibilidade 12

2.3.3.2 Exatidão 13

2.3.3.3 Precisão 13

2.3.4 Configurações robóticas 14

2.3.4.1 Configuração braço articulado 14

2.3.4.2 KUKA - KR 16-2 14

2.3.5 Comunicações industriais 16

2.3.5.1 DeviceNet 17

2.3.6 Segurança 17

2.4 Controlador Lógico Programável 19

2.4.1 O que é um Controlador Lógico Programável? 19

2.4.2 Arquitetura do Controlador Lógico Programável 20

2.4.3 CPU 21

2.4.4 Memória 22

2.4.5 Unidades de Entrada e Saída (E/S) 22

2.4.6 Terminal de programação . . . 23
2.4.7 Norma IEC 61131 . . . 24
2.4.8 Linguagens de programação para PLC . . . 24
2.4.9 Linguagem *Ladder* . . . 25
2.4.10 *Human Machine Interface*- HMI . . . 26

3 Trabalho Desenvolvido 28

3.1 Análise da primeira solução encontrada . . . 28
3.1.1 *Layout* da primeira solução . . . 29
3.1.2 Quadros completos . . . 29
3.1.3 Verificação dos esquemas e do material existente . . . 30
3.1.4 Implementação da primeira solução . . . 30
3.1.5 Transportador de entrada . . . 31
3.1.6 Transportador de saída . . . 32
3.1.7 Alimentador de placas . . . 32
3.1.8 Mesa de indexação . . . 33
3.1.9 Mesa de fixação . . . 34
3.1.10 Controladores, robô 1 e robô 2 . . . 35
3.1.11 *Gripper* robô 1 . . . 36
3.1.12 *Gripper* robô 2 . . . 37
3.1.13 Quadro elétrico . . . 38
3.1.14 PLC *OMRON CJ2M-CPU32* . . . 38
3.1.15 Consola *HMI OMRON nb10w-tw01b* . . . 39
3.2 Diagrama do sistema . . . 39
3.3 Comunicação *DeviceNet* . . . 41
3.3.1 Ligações físicas . . . 43
3.3.1.1 Unidade CJ1W-DRM21- Módulo *DeviceNet* . . . 44
3.3.1.2 Unidade BK5250 - Mesa de indexação . . . 45
3.3.1.3 Unidade KR - R1 e R2 . . . 45
3.3.2 Configurar rede . . . 45
3.3.2.1 Configurar a rede no *CX-Integrate* . . . 45
3.3.2.2 Configurar a rede no *WorkVisual* . . . 46
3.4 Ligações físicas . . . 47

3.5 Melhorias na mesa de indexação . . . 48
3.6 Melhorias na mesa de fixação . . . 50
3.6.1 Alinhamento do quadro . . . 51
3.6.2 Testes de aparafusamento . . . 53
3.6.3 Funcionamento das aparafusadoras . . . 54
3.6.4 Amolgar . . . 55
3.6.5 Melhorias . . . 57
3.6.6 Nova sequência lógica . . . 58
3.7 Programação Robô 2 . . . 58
3.8 Programação *PLC* . . . 63
3.8.1 *Function Blocks* . . . 66
3.8.2 Estrutura do programa principal - *Main* . . . 67
3.9 Programação HMI . . . 68
3.10 Otimização da célula . . . 75
3.10.1 Análise da solução otimizada . . . 76
3.10.2 Análise dos componentes de simulação . . . 76
3.10.3 Simulação Robô 1 . . . 77
3.10.4 Simulação Robô 2 . . . 78
3.10.5 Análise da solução otimizada final . . . 80

4 Resultados Experimentais **81**
4.1 Validação de componentes . . . 81
4.1.1 Testes na mesa de indexação . . . 81
4.1.2 Testes na mesa de fixação . . . 82
4.2 Testes de tempo de ciclo . . . 84
4.2.1 Tempo real - Primeira solução . . . 85
4.2.2 Tempo simulação - otimizado . . . 85
4.2.3 Comparação de tempos de ciclo . . . 86

5 Conclusão **88**
5.0.1 Trabalhos futuros . . . 89

Bibliografia **90**

Anexo A KUKA - KR 16-2 **93**

Anexo B KUKA - Sinais **95**

Anexo C HMI NB Series **101**

Apêndice A Linguagem Ladder **105**

A.0.1 Instruções do tipo Relé 105

A.0.2 Instruções de temporização e contagem 107

Apêndice B Circuito Pneumático **109**

Apêndice C Lista de Material **122**

Apêndice D Sequência Mesa Indexação **124**

Apêndice E Sequência Inicial Da Mesa De Fixação **126**

Apêndice F Erros Na Primeira Solução **128**

Apêndice G Sequência Final Da Mesa De Fixação **132**

Apêndice H Programação Robô 2 **134**

Apêndice I Programação *HMI* **142**

Apêndice J Otimização Da Célula **144**

Apêndice K Testes Na Mesa De Fixação **148**

Lista de Abreviaturas

ALU	Arithmetic Logic Unit
CAN	Controller Area Network
CIP	Common Industrial Protocol
CPU	Central Processing Unit
CTD	Counter Down
CTU	Counter Up
E/S	Entradas e Saídas
EDS	Electronic Data Sheet
EPROM	Erasable and Programmable Read Only Memory
FBD	Function Block Diagram
HMI	Human Machine Interface
IEC	International Electrotechnical Commission
IL	Instruction List
IP	Internet Protocol
ISO	International Organization Standardization
KRL	KUKA Robot Language
KUKA	Keller Und Knappich Augsburg
LCD	Liquid Crystal Display
LD	Ladder Diagram
LIN	Linear
ODVA	Open Devicenet Vendors Association

PC	Personal Computer
PLC	Programmable Logic Controller
PTP	Point-To-Point
RAM	Random Access Memory
ROM	Read Only Memory
SFC	Sequential Function Chart
ST	Structured Text
TOF	Timer Off Delay
TON	Timer On Delay
TP	Timer Pulse

Lista de Figuras

2.1 Estimativa anual mundial de fornecimento de robôs industriais 7
2.2 Primeiro robô industrial 9
2.3 Controlador KUKA e *Teach Pendant* 10
2.4 Componentes de um robô industrial articulado 11
2.5 Tipos de Juntas Utilizadas em Robôs 12
2.6 Recision vs Accuracy 13
2.7 Configuração Braço Articulado 14
2.8 Robô KUKA kr 16-2 15
2.9 Consola Kuka smartPAD 15
2.10 Risco inerente aos robôs 18
2.11 Controlador Lógico Programável 19
2.12 Arquitetura dos autómatos programáveis 20
2.13 Tipos de sinais 21
2.14 Terminal de programação 23
2.15 Terminal de programação 25
2.16 Degrau num diagrama de escada 26

3.1 *Layout* da primeira solução 29
3.2 Quadro Completo 30
3.3 Implementação da primeira solução na *RobotSol* 31
3.4 Transportador de entrada 32
3.5 Alimentador de placas 33
3.6 Mesa indexação 34
3.7 Mesa de fixação 35
3.8 Controlador e robô 36
3.9 *Gripper* robô 1 37
3.10 *Gripper* robô 2 37
3.11 Quadro Elétrico 38
3.12 *PLC OMRON CJ2M-CPU32* 39
3.13 Diagrama de controlo 41

3.14 Arquitetura geral de comunicações do sistema 41
3.15 Conector *DeviceNet* 43
3.16 Ligações Conector DeviceNet 44
3.17 Configuração unidade CJ1W-DRM21 44
3.18 Configuração unidade BK5250 - Mesa indexação 45
3.19 Configuração *CX-Integrate* 46
3.20 Configuração *WorkVisual* 47
3.21 Erros e melhorias na mesa de indexação 49
3.22 Detalhes dos Componentes da Mesa de Fixação 50
3.23 Características da mesa de fixação 52
3.24 Resultado do alinhamento 53
3.25 Base da aparafusadora recuado 54
3.26 Resultado base da aparafusadora 54
3.27 Ponta da aparafasudara alterada 55
3.28 Resultado do alinhamento 57
3.29 Melhorias finais 58
3.30 Programa principal do robô 2 60
3.31 Subprograma *pick* do quadro completo 62
3.32 Subprograma *Place* 63
3.33 Evolução da estrutura do código do PLC 65
3.34 Estrutura do programa *Main* 68
3.35 Ecrã principal da consola *HMI* 69
3.36 Ecrã correspondente aos estados da célula 70
3.37 Ecrã correspondente à zona do transportador de saída 71
3.38 Evolução da estrutura do código do PLC 72
3.39 Ecrã correspondente à zona do transportador de saída 73
3.40 Ecrã correspondente aos alarmes ativos 74
3.41 Ecrã correspondente ao histórico de alarmes 75
3.42 *Grippers* atualizados 77
3.43 Solução encontrada para o tapete de entrada 78
3.44 Esquemático 3D da primeira simulação do robô 2 79
3.45 Alterações realizadas na simulação do robô 2 80

4.1 Diagrama de tempo de ciclo da célula com a primeira solução 85
4.2 Diagrama de tempo de ciclo da célula com a solução otimizada 86
4.3 Comparação entre tempos de ciclo 87

A.1 Início de Ramificação 105
A.2 Fim de Ramificação 105
A.3 Contacto normalmente aberto 106
A.4 Contacto normalmente fechado 106
A.5 Saída normalmente aberta 106
A.6 Saída normalmente fechada 107
A.7 Lógica ladder de um Timer 107
A.8 Lógica ladder de um contador 108
A.9 Timmers ou Temporizadores 108

F.1 Material desgastado 128
F.2 Testes Mesa fixação 129
F.3 Alinhamento do Quadro 129
F.4 Testes Amolgar 131

H.1 Trajetória do Robô 2 134
H.2 Sinais *standard KUKA* utilizados entre robô e *PLC* 141

I.1 Ecrã da janela de contactos 142
I.2 Ecrã da janela dos sinais do robô 2 142
I.3 Ecrã da janela de estatísticas 143
I.4 Ecrã da janela do alarme de painel perdido 143

J.1 *Layout* otimizado 144
J.2 Esquemático 3D transportador de entrada 144
J.3 Esquemático 3D mesas desatualizados 145
J.4 Alteração do *gripper* do robô 1 145
J.5 Esquemático 3D *pick* transportador de entrada 145
J.6 Esquemático 3D *place* mesa de indexação 146
J.7 Esquemático 3D da primeira simulação do robô 2 146
J.8 Esquemático 3D da simulação final do robô 2 147

Lista de Tabelas

C.1 Material utilizado no Armazém (Pilha de entrada de painéis) 122
C.2 Material utilizado no transportador de entrada 122
C.3 Material utilizado na mesa de fixação 122
C.4 Material utilizado na mesa de indexação 123
C.5 Material utilizado no *gripper* do robô 1 123
C.6 Material utilizado no *gripper* do robô 2 123

H.1 Programas utilizados no robô 2 134
H.2 Áreas utilizadas pelo robô 2 134
H.3 Programas finalizados pelo robô 2 135
H.4 Áreas de colisão do robô 2 135
H.5 Lista de erros do robô 2 135

K.1 Teste em série com as bases da aparafusadora recuada 148
K.2 Teste em série com as bases da aparafusadora avançadas 148
K.3 Binário das aparafusadoras no nível 4 (13 voltas) 148
K.4 Binário da aparafusadora direita com 15 voltas 148
K.5 Binário da aparafusadora esquerda com 15 voltas 149
K.6 Binário da aparafusadora esquerda com 20 voltas 149

Capítulo 1

Introdução

Esta dissertação surge no âmbito do Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores da Universidade do Minho, com o objetivo de documentar o trabalho desenvolvido durante o projeto de dissertação, denominado "célula automatizada para montagem de quadros brancos com perfil de alumínio". A presente introdução pretende expor o problema que está na origem deste trabalho, assim como a sua contextualização, a motivação pessoal para a realização do mesmo e os objetivos que se propõem alcançar no final desta dissertação. Seguidamente, são enunciadas as tarefas propostas e o plano de trabalho.

1.1 Contextualização

Esta dissertação propõe retificar um sistema robotizado de montagem automático para substituição de linhas manuais tradicionais. Esta linha de montagem tradicional retrata uma montagem manual de quadros brancos, quadros de escrita que podem ser encontrados em escolas. Estes são constituídos por peças de plástico (topo e base) e perfis de alumínio (laterais) que formam a estrutura externa e por uma placa branca que forma o interior do quadro. O quadro completo é formado pela placa branca e pela estrutura externa, que é montado e aparafusado manualmente. Constata-se que é um trabalho repetitivo, aborrecido e com pouca produtividade.

No processo de automatizar esta linha manual, foi prevista uma solução para otimizar os ganhos de eficiência e qualidade. Assim, em 2015, foi iniciado o processo de otimização da linha. Esta solução previa a utilização de dois robôs *KUKA* (Kr 16-2), de duas mesas para montagem e aparafusamento dos quadros, de um transportador de entrada de acessórios, de uma pilha de entrada de painéis e de um transportador de saída.

O primeiro robô posiciona-se no transportador de entrada de acessórios, onde se encontram as peças de plástico e os perfis de alumínio, agarra numa peça de cada vez e, coloca na posição correta na mesa de montagem. Por último, agarra num painel que se encontra na pilha de entrada de painéis e coloca-o na mesa de montagem. De seguida, ocorre a montagem do quadro e, quando concluída, o segundo robô situa o quadro completo na mesa de fixação, onde se dá o aparafusamento do mesmo e onde se amolgam os perfis de alumínio, terminando assim o processo.

Visto que o projeto começou em 2015, foi sofrendo alterações e melhorias, mas nunca foi possível produzir grandes quantidades de quadros com um erro admissível, isto é, que satisfizessem a produção exigida pelo cliente. Assim, nesta dissertação, será retificada parte da solução prevista, que envolve um robô e a mesa de aparafusamento. Por ser um projeto antigo com muitas alterações e com documentos desatualizados, decidiu-se não utilizar o trabalho já anteriormente realizado (código de robô, *PLC*, *HMI*) e começar o projeto apenas com o material disponível. Assim, será uma forma de aprendizagem e atualização de documentos.

Esta dissertação foi desenvolvida na empresa *RobotSol*- Engenharia Industrial, SA, que é uma empresa sediada na Maia e que se dedica ao desenvolvimento de soluções à medida, de equipamentos e linhas para a indústria em geral. Um dos objetivos é poupar o máximo possível em componentes, utilizando apenas os materiais existentes na empresa.

A escolha de fazer a dissertação numa empresa teve por base o facto de ter maior possibilidade de aquisição de conhecimento e experiência numa área com grande saída profissional. Outro motivo, é o facto de os robôs poderem fornecer o conforto necessário ao ser humano, para que este não tenha que executar trabalhos pesados, perigosos ou até que possa deixar de trabalhar.

Após uma explicação do problema, apresentam-se os objetivos deste projeto, sendo que o principal objetivo é o funcionamento correto da célula com um erro admissível, isto é, que satisfaça a produção exigida pelo cliente (2 a 2,5 quadros/min). Assim, é necessário estudar a solução existente e explorar as suas capacidades de produção e necessidades para que a mesma seja otimizada. Podem-se resumir os restantes objetivos como:

- Verificar o *hardware* da célula;
- Desenvolver o código para o robô (trajetórias, aviso de erros e comunicação com *PLC*);
- Desenvolver o código para o *PLC* (controlo da mesa de fixação, aviso de erros, comunicação com o robô, comunicação com o *HMI*);
- Desenvolver o software necessário para o *HMI* (aviso de erros, comunicação com o *PLC*);
- Otimizar a solução existente e simular a solução em 3D;
- Implementar a solução proposta no cliente.

1.2 Proposta de Trabalho e Descrição das Tarefas

Após uma análise do problema existente, optou-se por dividi-lo em seis partes. A primeira parte foi dedicada à verificação do material existente, isto é, confirmar o que existia na empresa *Robotsol* e aprender o seu funcionamento. A segunda parte, visou a programação do robô, isto é, a marcação dos pontos de trajetória que foram percorridos pelo mesmo. A terceira parte foi dedicada ao controlo da mesa, bem como aos sinais do robô, através do controlador lógico programável (*PLC*). A quarta parte focou-se na programação da consola. A quinta parte teve por base a simulação 3D de toda a célula, bem como as otimizações necessárias à primeira parte da solução para que o tempo de ciclo fosse o menor possível. Por fim, a sexta parte focou-se na implementação da solução no cliente. De seguida, dividiu-se o problema em várias tarefas:

- **Tarefa 1** - Estudo do tema e estado da arte: pesquisa, análise e estudo bibliográfico, com escrita do estado da arte, relativamente ao tema proposto.
- **Tarefa 2** - Análise do sistema e arquitetura de software: analisar o sistema, bem como todos os componentes necessários para o seu funcionamento. Dessa maneira, foi possível criar um fluxograma com o funcionamento do sistema.
- **Tarefa 3** - Programação do robô offline: desenvolvimento do algoritmo a utilizar, que permitiu controlar os sinais necessários e pontos (trajetórias) para o funcionamento do robô.
- **Tarefa 4** - Teste da programação realizada e afinações necessárias.
- **Tarefa 5** - Controlo (*PLC*) da mesa de aparafusamento e do tapete de saída: desenvolvimento do algoritmo que permitiu a sequência correta da mesa de aparafusamento e da saída.
- **Tarefa 6** - Teste da programação executada e afinações necessárias.
- **Tarefa 7** - Desenvolvimento do algoritmo que permitiu a comunicação correta entre *PLC* e robô.
- **Tarefa 8** - Ciclo automático do sistema (automático externo): ligação do robô com o *PLC* e criação de um sistema automático e funcional.
- **Tarefa 9** - Teste da programação feita e afinações necessárias.
- **Tarefa 10** - Desenvolvimento do algoritmo do *HMI* e comunicação correta entre *HMI* e robô.
- **Tarefa 11** - Afinações e testes na empresa: validação do sistema e afinações necessárias.

- **Tarefa 12** - Simulação 3D da célula e otimização da mesma.
- **Tarefa 13** - Atualização do algoritmo à nova realidade (*PLC*, robô, *HMI*).
- **Tarefa 14** - Teste do sistema otimizado e afinações necessárias.
- **Tarefa 15** – Implementação no cliente.
- **Tarefa 16** - Afinações finais no cliente: afinações dos sensores e melhoramento das trajetórias.
- **Tarefa 17** - Escrita da dissertação.

1.3 Organização da dissertação

Esta secção é um resumo de todos os capítulos presentes na dissertação.

A dissertação é composta por um total de cinco capítulos, dos quais o primeiro apresenta uma introdução à dissertação, o segundo retrata o estado da arte, o terceiro aborda o trabalho desenvolvido, o quarto demonstra os resultados experimentais e os seus resultados e, por fim, no capítulo 5 são apresentadas as conclusões à dissertação e possíveis melhorias.

No capítulo 1 são realizadas uma contextualização do problema e uma introdução ao sistema bem como apresentados os objetivos propostos. São ainda referidos os trabalhos propostos e a descrição das tarefas. De seguida, no capítulo 2 são abordados a história relativa à robótica industrial bem como um estudo teórico sobre conceitos importantes na robótica. Também são observados casos de estudo para aprendizagem e redução dos tempos de ciclo. No capítulo 3 inicia-se a análise da primeira solução. De seguida são efetuadas uma descrição do sistema, bem como a configuração da comunicação utilizada. Após isto, explicam-se as ligações físicas utilizadas e as melhorias realizadas nas mesas de indexação e fixação. Depois, descreve-se o código desenvolvido para o robô 2, o *PLC* e a *HMI*. Por fim, criou-se o *layout* na simulação, otimizando o máximo possível a célula. No capítulo 4 são descritos os testes realizados na célula, demonstrados os tempos de ciclo da primeira solução e da solução otimizada e é apresentada uma comparação entre esses tempos. Finalmente, no capítulo 5 são apresentadas as conclusões da dissertação. Neste capítulo também são referidos os possíveis trabalhos futuros relativamente ao projeto.

Capítulo 2

Estado da Arte

A robótica abrange várias áreas tecnológicas, nomeadamente, a eletrónica, a mecânica e a computação. Tendo em conta os objetivos propostos, neste capítulo, são analisados casos reais da utilização de robôs, bem como o robô utilizado neste projeto. Relativamente ao robô, são estudadas as suas características permitindo avaliar o seu funcionamento e poder de execução. São, também, estudados os fundamentos teóricos necessários para o desenvolvimento deste projeto.

2.1 Casos de estudo

De forma a encontrar a melhor solução possível para o problema proposto, foi feito um estudo de vários casos reais de otimizações de linhas de produção, desde a colocação dos vários componentes referentes a essa linha, até ao estudo do trajeto executado pelo robô. O objetivo principal de uma linha de produção é a sua produtividade elevada com custos reduzidos, para que haja a maior percentagem de lucro possível.

2.1.1 1º caso de estudo

No primeiro caso de estudo, foi realizado o estudo da automatização das operações de montagem relacionadas com as operações manuais de montagem e movimentação de material na célula. Este projeto desenvolveu-se segundo uma metodologia baseada nos princípios *Lean*, em que se pretendeu reduzir o tempo de ciclo (*cycle time*) das operações através da redefinição do *layout* de maneira a haver um fluxo de produção contínuo. Para tal, pretendeu-se uma produção de lote unitário (*one-piece flow*) que visou prevenir a ocorrência de filas de espera e consequente atraso na produção, diminuindo os tempos de ciclo [1].

Assim, este caso demonstrou que o *layout* geral do problema descrito desta dissertação tem que ter o material necessário para o funcionamento da célula próximo do robô, para uma fácil utilização do mesmo e, na sua saída, uma fluidez constante para não existir paragem na produção.

2.1.2 2º caso de estudo

No segundo caso, o principal objetivo relacionou-se com a necessidade de desenvolver estratégias de planeamento de trajetórias de um robô e aproveitar cada estratégia em ambientes industriais. É importante a sua utilização em abordagens de *pick-and-place*, dado que, para cada trajetória, existem muitas configurações possíveis [2].

Assim, este caso demonstrou que se tem que ter em atenção as trajetórias e configurações escolhidas, nunca esquecendo os cenários envolventes ao robot. Por fim, com um estudo da trajetória e do meio envolvente ao robô, o tempo de ciclo (*cycle time*) diminui.

O principal objetivo deste projeto é o desenvolvimento e implementação de um planeamento de trajetória em manipuladores e em ambiente industrial, sem nunca esquecer os passos intermédios inerentes a este processo, nomeadamente o reconhecimento do cenário envolvente ao robot.

2.1.3 3º caso de estudo

No terceiro caso de estudo, o objetivo da dissertação de mestrado consistiu no controlo de rolhas defeituosas que não podiam ser comercializadas. O processo de recolha foi feito manualmente, tendo sido necessário desenvolver um sistema de recolha automático das rolhas, com recurso a um manipulador de seis graus de liberdade. A tarefa consistiu numa operação de *pick-and-place* das rolhas que se encontravam em movimento num tapete automático. Então, realizou-se um estudo de vários algoritmos de planeamento de movimento, concretamente de trajetória e de caminho [3].

Assim, mesmo que esta dissertação não tenha sido implementada fisicamente, concluiu-se que uma das maiores preocupações a ter em conta é a do tempo perdido em trajetórias, uma vez que levou a uma maior produção.

2.2 História da robótica

Desde há muito tempo que a Humanidade tem vindo a criar máquinas para auxiliar na execução de diversas atividades quotidianas. Naturalmente, surgiu um novo termo, *Robô (Robot)* palavra eslava que significa trabalhador [4].

Inicialmente, essas máquinas eram muito rudimentares: os Egípcios fizeram braços mecânicos para imitar os seus Deuses; os Gregos usaram estátuas hidráulicas para demonstrar a ciência hidráulica; os Europeus, criaram fantoches mecânicos para diversão da população. Como se pode constatar destes

exemplos, não era possível programá-los nem utilizá-los de forma a realizar trabalho, visto que apenas executavam uma tarefa. Em 1921, através do romance teatral *"Rossum's Universal Robots"*, no qual *Karel Kapec* retratou os robôs como máquinas de trabalho incansáveis e de aspeto humano, surgiu o termo *Robot* que só foi popularizado em 1950 devido ao livro *"I, Robot" (Eu, Robô)* de *Isaac Asimov* [5].

Desta forma, com o desenvolvimento tecnológico, foi possível construir robôs aptos para executarem várias tarefas, que realizam trabalho e com capacidade de serem programáveis. Várias são as áreas que, hoje em dia, recorrem a serviços robotizados. Por exemplo, podem destacar-se áreas como a indústria, brinquedos, agricultura, segurança, serviços, entre outras. Para uma visualização mais concreta sobre a utilização dos robôs, estes são usados na indústria para soldagem, nas linhas de montagem, para pintura, entre outras aplicações. Pode concluir-se que a introdução de robôs é uma mais valia para a Humanidade, pois estes executam tarefas repetitivas, aborrecidas e perigosas, trabalhando as horas que forem necessárias para aumentar a produtividade, com maior precisão e velocidade, deixando o ser humano repousado [6].

Neste trabalho, é destacada a área da indústria, principalmente a linha de montagem, pois enquadra-se no tema da dissertação. Para além disso, torna-se relevante referenciar que os robôs têm um papel muito importante na indústria, sendo que, em 2016, foram vendidas 1828 mil unidades de robôs (um aumento de 12% em relação ao ano anterior) prevendo-se um aumento nos anos seguintes, Figura 2.1.

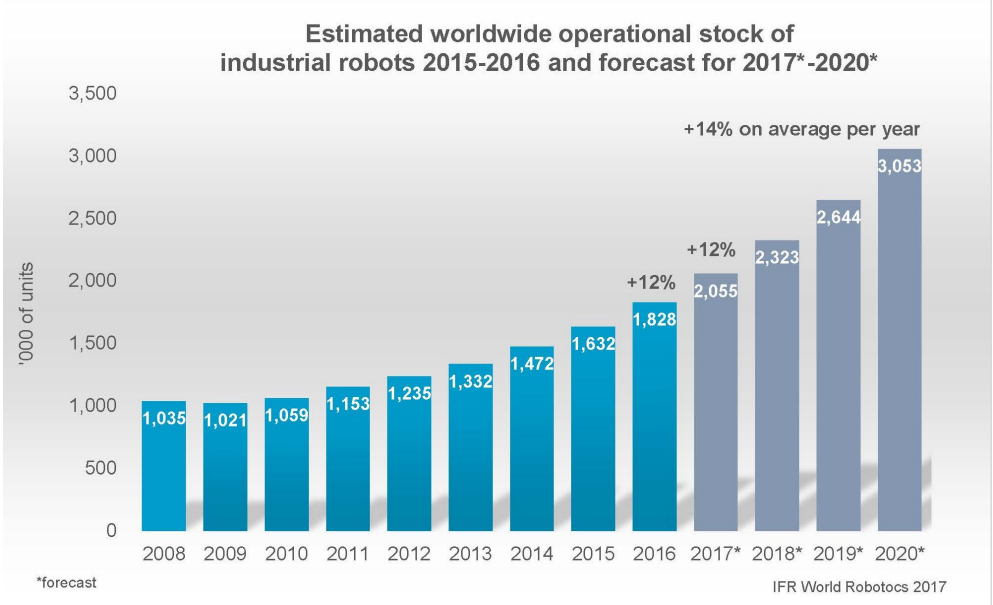


Figura 2.1: Estimativa anual mundial de fornecimento de robôs industriais [7]

2.3 Robôs industriais

A história da robótica é rica em criatividade cinematográfica, engenho científico e uma visão empreendedora. Pode dizer-se que é uma mistura de fantasia com inspiração, sendo que é difícil definir o que é um robô. Pela definição ISO 8373, *International Organization for Standardization*, um robô é *"um manipulador controlado, reprogramável, com múltiplos propósitos e constituído por três ou mais eixos, que pode ser fixo ou móvel, sendo utilizado em aplicações de automação industrial"*. A *Robot Institute of America* designa um robô como *"reprogramável, um manipulador multifuncional designado para mover materiais, partes, ferramentas ou dispositivos especializados através da programação de movimentos"*.

A necessidade de aumentar a produção e reduzir os custos da mesma, levou Henry Ford, no século XX, a inventar um método de produção por linhas de montagem em série [8].

Com o desenvolvimento tecnológico começaram a utilizar-se robôs no mundo industrial. Na data de 1961, foi criado o primeiro robô da marca *UNIMATE*,Figura 2.2. Foi inserido numa fábrica da *General Motors em Trenton*, Nova Jérsia, onde descarregou peças de alta temperatura de uma máquina de fundição, trabalho difícil e pouco popular [9].

Na Europa, um pouco mais tarde, em 1967, foi instalado um robô da *UNIMATE* na *Metallverken*, Suécia [4]. Desde então, os robôs sofreram uma grande evolução tendo, atualmente, boas características de repetibilidade e precisão de movimento.

Assim, os robôs começaram a ser utilizados em várias tarefas, nomeadamente:

- Tarefas repetitivas, aborrecidas e sem significado.
- Tarefas pesadas e difíceis.
- Ambientes perigosos [5].

Figura 2.2: Primeiro robô industrial [6]

2.3.1 Componentes de um robô industrial

Existem diversos robôs industriais com estruturas, formas e funções diferentes, sendo, no entanto, possível identificar elementos comuns entre os mesmos, Figura 2.4. De seguida, são apresentados esses elementos:

I **Manipulador mecânico:** Menciona a estrutura e aspeto físico do robô, muitas vezes referido como o "braço" do robô, é composto por vários elementos estruturais rígidos (elos) ligados através de articulações (juntas), formando uma cadeia cinemática. A primeira estrutura é designada base e a última *flange*, onde será ligada a extremidade terminal (ex. garra).

a) **Atuadores:** São os componentes responsáveis por converter energia elétrica, hidráulica ou pneumática, em potência mecânica. Através dos sistemas de transmissão, essa potência mecânica gerada pelos atuadores é enviada aos elos para que os mesmos se movimentem.

b) **Sensores:** Fornecem informações do manipulador, isto é, do modo de interação entre o robô e o ambiente (força, torque, sistema de visão) à unidade de controlo. Também fornecem informações como a posição e a velocidade dos elos.

II **Controlador:** É o "cérebro" do robô industrial. É uma espécie de computador com funções particulares e responsável por gerir e monitorizar os parâmetros necessários para realizar as tarefas do robô. Tem a possibilidade de comunicação através de sistemas de barramento (ex. *ProfiNet, Ethernet, Interbus*), *PLC* e comunicação através da rede (ex. computador central, *service Notebook*,

outras unidades de comando). Juntamente com o controlador, existe uma consola portátil (*Teach Pendant*) que permite a interação entre o robô e o utilizador, Figura 2.3. Este aparelho permite algumas funcionalidades como a programação de um robô, a memorização de posições do robô, a verificação do estado do manipulador, controlo de sinais e a movimentação do mesmo. A linguagem de programação é específica de cada fabricante e pode ser feita *on-line* (na consola) ou *off-line* (no computador).

Figura 2.3: Controlador KUKA e *Teach Pendant* [11]

III **Unidade de potência:** Responsável pelo fornecimento de energia aos atuadores [10].

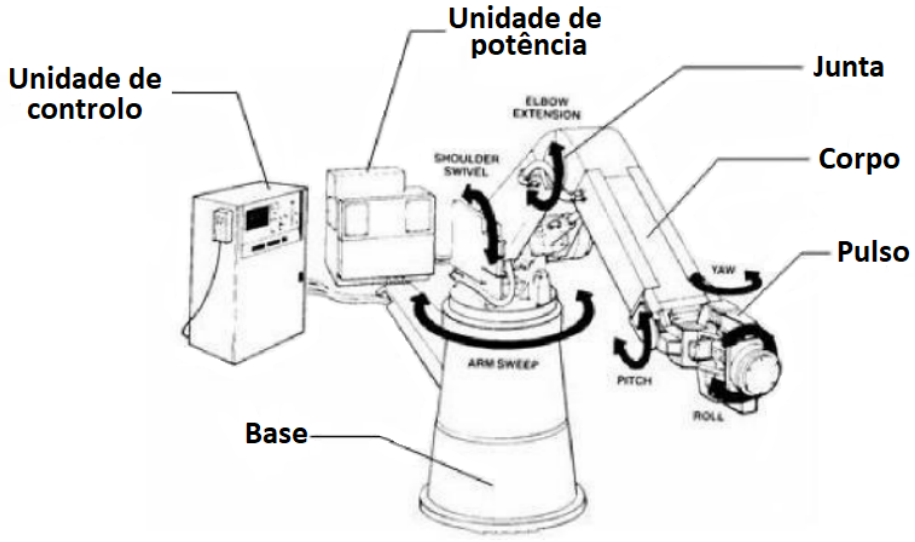
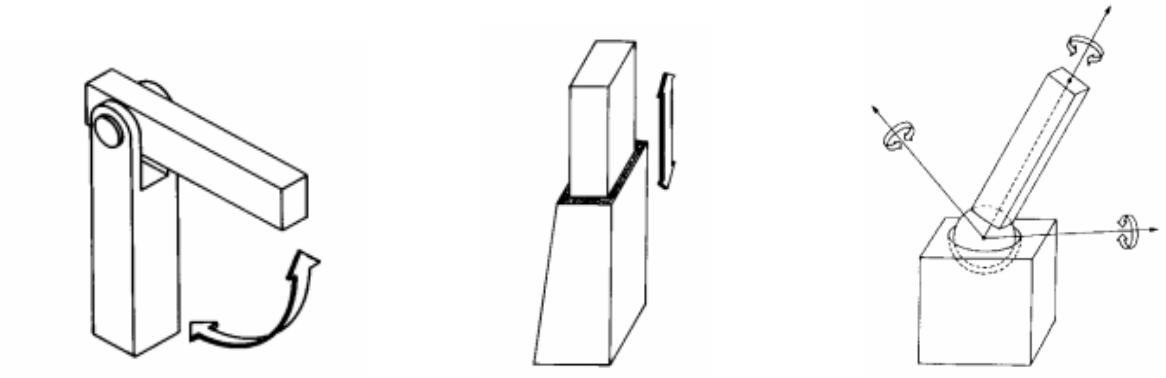


Figura 2.4: Componentes de um robô industrial articulado [10]

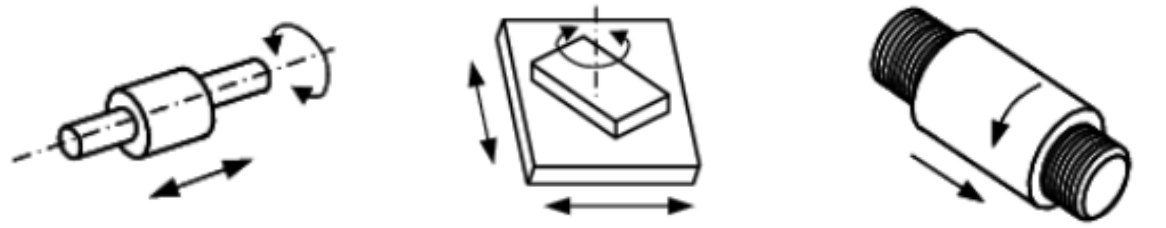
2.3.2 Juntas

Existem vários tipos de juntas, nomeadamente, rotativas, prismáticas, esféricas, cilíndricas, planares e de parafusos. Dependendo de cada tipo, estas podem-se mover numa, duas ou três direções, ou graus de liberdade. Em seguida, são descritas as suas funcionalidades.

- **Rotacionais (R)** – O movimento relativo dos elos é rotacional, isto é, gira em torno de uma linha estacionária (eixo de rotação), Figura 2.5a.
- **Translação ou Prismática (P)** – O movimento relativo dos elos é linear, isto é, move-se em linha reta, Figura 2.5b.
- **Esférica (E)** – Combinação de três juntas de rotação, permitindo rotações em torno de três eixos distintos, Figura 2.5c.
- **Cilíndrica (C)** – Composta por duas juntas, uma rotacional e uma prismática, Figura 2.5d.
- **Planar** – Composta por duas juntas prismáticas, realiza movimentos em duas direções, Figura 2.5e.
- **Fuso ou Parafuso (H)** – Constituída por um parafuso e uma rosca que executa um movimento semelhante ao da junta prismática, mas com movimento de rotação no eixo central, Figura 2.5f [12].



(a) Rotacional (R) **(b)** Translação ou Prismática (P) **(c)** Esférica (E)



(d) Cilíndrica (C) **(e)** E-Planar **(f)** Fuso ou Parafuso (H)

Figura 2.5: Tipos de Juntas Utilizadas em Robôs [12]

2.3.3 Desempenho do robô

O desempenho do robô industrial é especificado em termos de operações funcionais e tempo de ciclo. Por exemplo, robôs de montagem têm especificação por ciclos de "*pick-and-place*" por minuto.

2.3.3.1 Repetibilidade

Esta especificação quantifica a capacidade de o manipulador retornar para o mesmo local nas mesmas condições de repetibilidade, isto é, corresponde ao número de locais (o raio de uma esfera) que o manipulador retorna quando enviado da mesma origem, com as seguintes condições de repetibilidade [6]:

- O mesmo procedimento de medição ou procedimento de teste;
- O mesmo operador;
- O mesmo equipamento de medição ou teste utilizado nas mesmas condições;
- O mesmo local;
- Repetição por um curto período de tempo [13].

2.3.3.2 Exatidão

Grau de concordância entre a expectativa de um resultado de teste e um valor verdadeiro, ou entre um resultado de medição e um valor verdadeiro, Figura 2.6.

- A "exatidão de medição" não é uma grandeza e não lhe é atribuído um valor numérico. Quando o erro de medição é menor, a medição em si é mais exata.
- Não é recomendado utilizar o termo "exatidão de medição", mas sim "veracidade de medição".
- Na prática, o valor de referência aceite é substituído pelo valor verdadeiro [14].

2.3.3.3 Precisão

Esta especificação retrata a capacidade que um robô tem para posicionar a sua extremidade terminal num local pré-programado no espaço [6], isto é, o grau de concordância entre os resultados dos testes independentes obtidos sob condições estipuladas e o valor de referência, figura 2.6

- A precisão depende apenas da distribuição de erros aleatórios e não está relacionada com o valor verdadeiro ou com o valor especificado.
- A medida de precisão é, geralmente, expressa em termos de imprecisão e calculada como um desvio padrão dos resultados do teste, sob condições especificadas de medição.
- A precisão de medição é utilizada para definir a repetibilidade de medição, a precisão intermediária de medição e a reprodutibilidade de medição [15].
- As condições especificadas podem ser condições de repetibilidade, condições de precisão intermediária ou condições de reprodutibilidade [16].

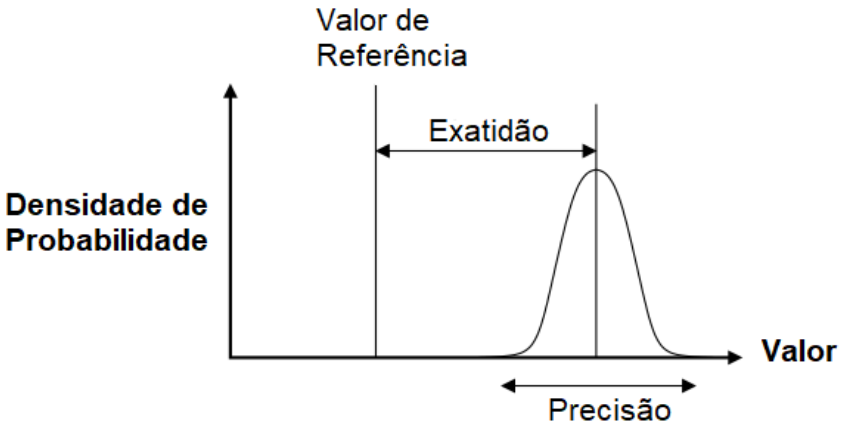


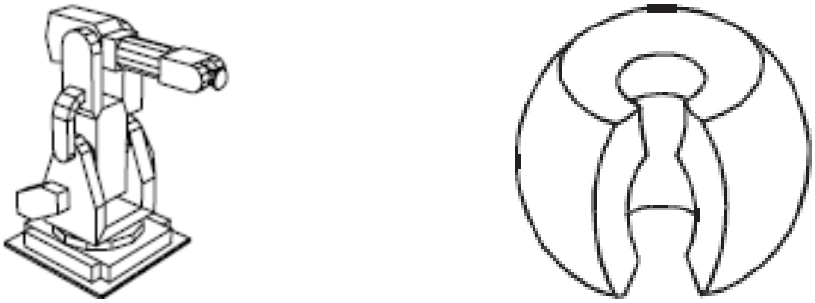
Figura 2.6: Recision vs Accuracy [5]

2.3.4 Configurações robóticas

Posto isto, é feita uma análise sobre os diferentes tipos de configurações de robôs, mostrando as vantagens e desvantagens e quais as suas aplicações.

2.3.4.1 Configuração braço articulado

A configuração braço articulado é a mais usada na indústria, por ser semelhante a um braço humano (ombro, cotovelo e pulso) e possuir três juntas rotativas, todas elas rotacionais. O pulso é unido na extremidade do antebraço, o que facilita a orientação do órgão terminal. Possui um volume de trabalho semelhante à configuração de um braço humano e têm uma área de trabalho maior em relação ao tamanho do robô. O seu modelo cinemático é complexo, tendo uma estrutura não muito rígida nos limites do braço, o que faz com que seja necessário um técnico especializado para o programar. Esta configuração é bastante utilizada em tarefas de manuseio de material, soldadura, pintura e paletização [9].



(a) Braço robótico **(b)** Área de trabalho

Figura 2.7: Configuração Braço Articulado [9].

2.3.4.2 KUKA - KR 16-2

O robô utilizado nesta dissertação é um robô KUKA muito comum na indústria, o KR 16-2 Figura 2.8.

Para a sua escolha, foram tidos em consideração vários parâmetros, nomeadamente, repetibilidade, velocidade, aceleração e número de eixos. Este robô tem seis eixos, carrega até 16Kg, sendo que o seu espaço de trabalho está representado no anexo A. Tem repetibilidade e velocidade altas, uma precisão de repetição 0.015mm - 0.2mm, um elevado tempo de vida e o seu controlador é *KR C4*.

Figura 2.8: Robô *KUKA* kr 16-2 [17]

A linguagem de programação utilizada é *KRL - KUKA Robot Language*. O robô é programado *online* ou *offline*. Na programação *online*, o processo é o *Teach-in* e utiliza-se a consola *KUKA smartPAD*, retratada na Figura 2.9. Na programação *offline*, o processo é programado por interação gráfica (simulação com o programa *KUKASim*) ou programação por texto (Utiliza um PC com o programa *WorkVisual*).

Figura 2.9: Consola Kuka smartPAD [18]

Por fim, existem quatro tipos de operação de um robô *KUKA*:

- T1(Velocidade reduzida manual):

  Para modo de teste, programação e aprendizagem ("*Teach*");

  Velocidade na operação do programa de no máximo 250 mm/s;

  Velocidade na operação manual de no máximo 250 mm/s.

- T2 (Velocidade elevada manual):

  Para modo de teste;

  Velocidade na operação do programa correspondente à velocidade programada;

  Modo manual: não é possível.

- AUT (automático):

  Para robô industrial sem unidade de comando superior;

  Velocidade na operação do programa correspondente à velocidade programada;

  Modo manual: não é possível.

- AUT EXT (automático externo):

  Para robô industrial com unidade de comando superior (PLC);

  Velocidade na operação do programa correspondente à velocidade programada;

  Modo manual: não é possível [18].

2.3.5 Comunicações industriais

Inicialmente, na indústria, as redes de comunicação utilizadas eram otimizadas para aplicações específicas, normalmente, para um melhor controlo, uma melhor gestão e uma maior segurança e informação. Embora adequadas à funcionalidade para a qual foram projetadas, essas redes de comunicação não foram desenvolvidas tendo em mente uma única arquitetura corporativa coerente. Assim, devido à eficiência, fiabilidade e aos gastos associados à implementação de cada rede, os fabricantes foram forçados a implementar várias redes diferentes abrindo portas para um novo problema, de que nenhuma das redes comunicava com a outra. Como resultado, ao longo do tempo, a maioria dos ambientes de rede de empresas foram caracterizadas por inúmeras redes especializadas, geralmente incompatíveis, existentes num único espaço. Hoje, no entanto, devido à Internet é possível automatizar todo o processo e ligar toda a empresa. Assim, é obrigatório existir algum tipo de comunicação que permita a troca de informação. Para tal, surgem as comunicações industriais. Atualmente não existe nenhum protocolo de comunicações industriais comum em todos os processos fabris nas diferentes fábricas. Isto, faz com que existam vários protocolos de comunicações industriais, sendo as quatro melhores redes da *Open Devicenet Vendors Association* - ODVA (*EtherNet/IP*, *DeviceNet*, *ControlNet* e *CampoNet*). Estas comunicações es-

tão ligadas por um dos protocolos mais versáteis da automação industrial, nomeadamente o *Common Industrial Protocol* (CIP). Estas comunicações têm características que podem ir de encontro com os diferentes níveis de uma fábrica: administração, sistemas de trabalho (conjunto de células de trabalho) e células de trabalho [19].

Nesta subsecção é apresentado, de forma muito breve, o protocolo de campo utilizado nesta dissertação. Também são apresentadas algumas das suas características, assim como os seus princípios de funcionamento.

2.3.5.1 DeviceNet

No ano 1994 surgiu um protocolo de rede (*DeviceNet*) que permite que os dispositivos de controlo possam comunicar entre eles. A *DeviceNet* foi a primeira implementação do CIP e é baseada na rede de área do controlador (*Controller Area Network* – CAN). Esta, tem algumas características, nomeadamente:

- Suporte até 64 nós por rede;
- Alimentação de dispositivos até 21V e 8A;
- Inserção ou remoção de nós enquanto a rede está ativa;
- *QuickConnect* para dispositivos que são frequentemente removidos e adicionados à rede, por exemplo, ferramentas robóticas;
- Uso de conetores selados;
- Taxas de dados selecionáveis de 125, 250 e 500KBaud;

O protocolo de comunicação usado na camada de aplicação não é baseado no endereçamento de processos de aplicação, mas sim na definição de objetos de comunicação (orientado a objetos) [19].

2.3.6 Segurança

Os robôs são considerados máquinas potencialmente perigosas. Para evitar acidentes, o maior aspeto que se deve ter em conta é a segurança. O facto de os robôs operarem, frequentemente, com outras máquinas e seres humanos, faz com que haja um aumento da probabilidade de ocorrerem acidentes. Observando a Figura 2.10, repara-se que existe um risco associado à utilização dos robôs, sendo que um robô tem como possíveis perigos: o impacto provocado pelo seu movimento, queimaduras, choques elétricos, entre outros. Para evitar este tipo de acidentes, existem normas e regras internacionais e

nacionais. A norma ISO 10218:2011, indica métodos de segurança para a integração de robôs industriais, bem como orientações para o uso seguro dos robôs ou sistemas robóticos [20].

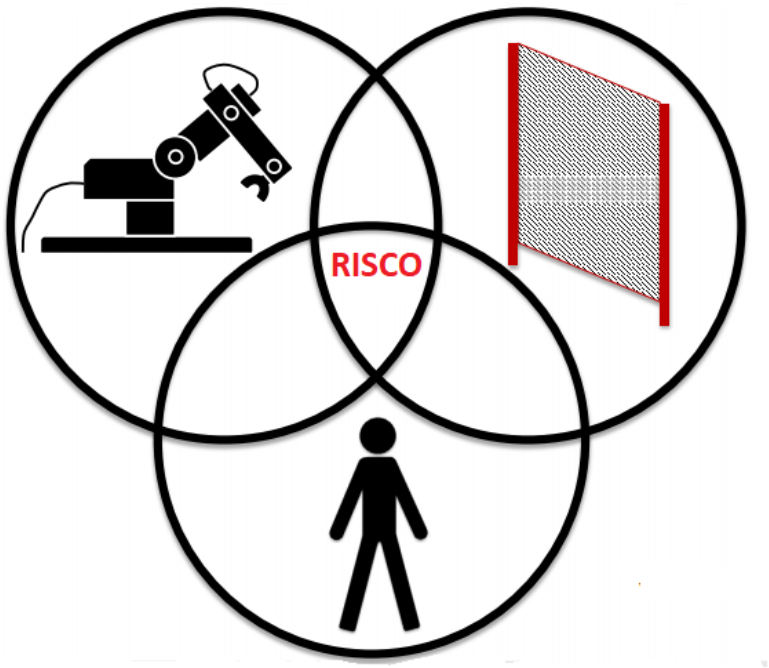


Figura 2.10: Risco inerente aos robôs [21]

Em Portugal, a diretiva 2006/42/CE [22], conhecida como "Diretiva Máquinas", existe para regulamentar a colocação no mercado e a entrada em serviço de máquinas novas, transposta para o Decreto-Lei n.º103/2008 [23].

Assim, quando existe interação entre um operador e uma célula onde está presente um manipulador, devem-se respeitar 3 regras. São elas:

1. Se o robô estiver parado, não se deve assumir que não se vai mover;
2. Se o robô repete um padrão de movimento, não se deve assumir que o vai continuar a repetir;
3. Ter atenção ao robô e ao que ele pode fazer [24].

De modo a reduzir o risco de acidentes, existem 2 tipos de medidas de proteção, as passivas e as ativas. As medidas de proteção passivas são aquelas que tentam prevenir os acidentes. Estas são: identificação de zonas de perigo; limitação do espaço de trabalho dos robôs ao mínimo necessário; regras de conduta implementadas pela empresa; formações de segurança aos funcionários que irão trabalhar diretamente com os robôs. As medidas de proteção ativas são medidas que limitam os efeitos ou consequências quando não se consegue evitar o acidente, a saber são: colocação de sensores nas zonas

perigosas; circuito de paragem e emergência; proteção contra colisões; sistema para impedir o acesso à área de trabalho dos robôs [20].

As fases nas quais existem mais acidentes são na de arranque e na de manutenção do sistema. Na fase de arranque, quando se encontra em modo automático e em operação, deve ser verificado o sistema e confirmar se toda a sua segurança está ativa. Na fase de manutenção e reparação, o sistema deve encontra-se desligado e devem ser seguidos todos os procedimentos fornecidos pela empresa responsável pelo mesmo.

2.4 Controlador Lógico Programável

Os controladores lógicos programáveis, conhecidos como autómatos ou PLC do inglês *"Programmable Logic Controllers"*, são computadores industriais constituídos por CPU e memória que, por serem robustos, são utilizados em ambiente industrial para controlo de máquinas e processos.

2.4.1 O que é um Controlador Lógico Programável?

O Controlador Lógico Programável (PLC), representado na Figura 2.11, é um microprocessador que usa memória programável para armazenar instruções e executar funções como temporização, contagem, lógica, entre outras, para controlar máquinas e processos [25].

O PLC, no seu programa de controlo, utiliza uma linguagem de programação simples e intuitiva. Assim, o PLC foi projetado para que qualquer engenheiro, ainda que com menos capacidade de programação, consiga configurar e alterar os programas. Com isto, não é necessário um programador especializado para realizar tal tarefa [26].

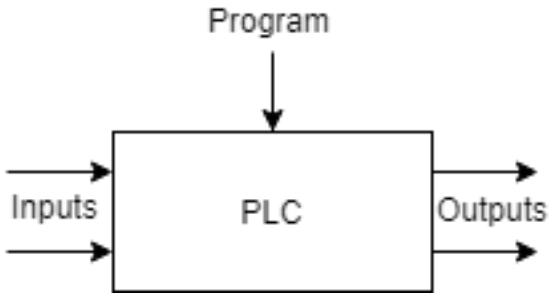


Figura 2.11: Controlador Lógico Programável

O termo 'lógico' é utilizado devido ao *PLC*, na sua programação, implementar lógica e operações de comutação, como por exemplo, se A ou B ocorre, então C é ligado. Os dispositivos de entrada (sensores,

interruptores, etc) e os dispositivos de saída (motores, válvulas, etc) do sistema são controlados e conectados ao *PLC*. De seguida, o controlador monitoriza as entradas e saídas e executa as regras de controlo de acordo com o seu programa. Uma grande vantagem da utilização de *PLC* é que o mesmo controlador básico pode ser usado para uma gama ampla de sistemas de controlo. Para modificar o sistema e as regras de controlo é apenas necessário utilizar instruções diferentes, isto é, programar o *PLC*, não sendo necessária a mudança de cabos. O resultado é um sistema flexível, barato e resistente, em que os *PLC's* foram otimizados para tarefas de controlo e ambiente fabril [26].

As vantagens da utilização do *PLC* são:

- Resistência, pois são projetados para suportar vibrações, variação de temperatura e humidade e ruído elevados, todas elas condições que se apresentam em ambiente fabril.
- Possuem interface para entradas e saídas, que estão dentro do controlador.
- São facilmente programados e possuem uma linguagem de programação acessível.

2.4.2 Arquitetura do Controlador Lógico Programável

Todos os controladores lógicos programáveis possuem a mesma arquitetura e os mesmos componentes básicos, como se pode verificar na Figura 2.12.

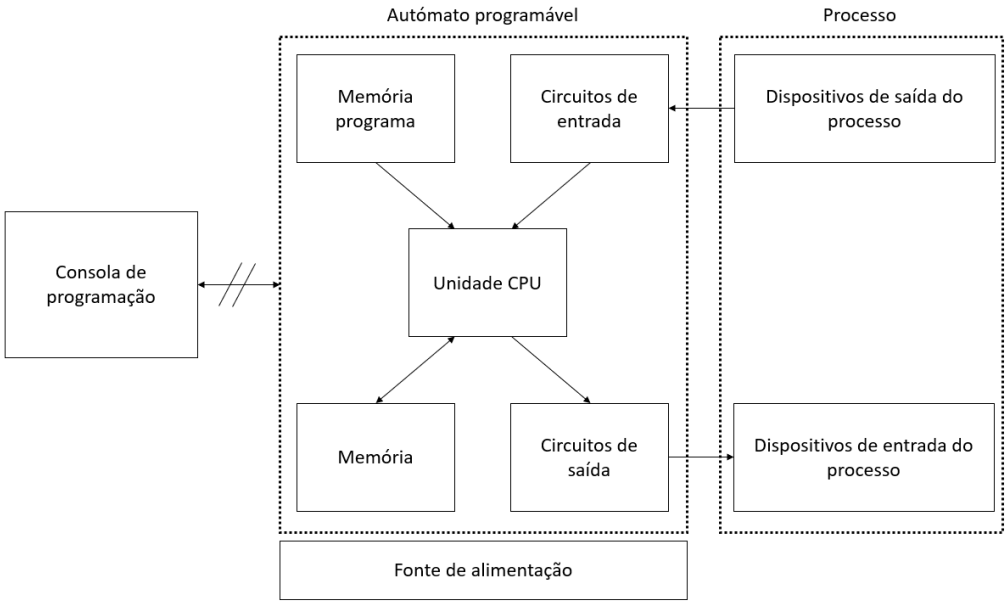
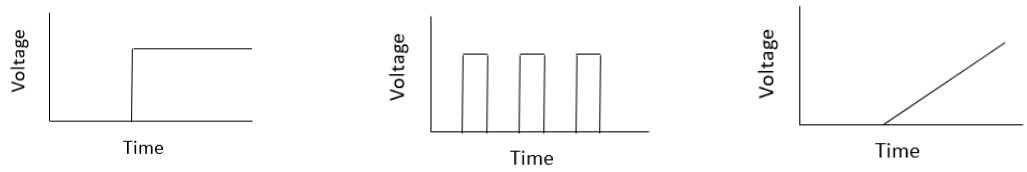


Figura 2.12: Arquitetura dos autómatos programáveis [27]

Podem ser considerados como componentes básicos, o processador, as memórias, os barramentos, os dispositivos de entrada e saída, a fonte de alimentação e o terminal de programação [26].

- A Unidade Central de Processamento (CPU) é a unidade que contém o microprocessador, que executa ciclicamente o código armazenado na memória do programa. Esta unidade recebe os sinais dos dispositivos de entrada e interpreta-os, executando as ações de controlo de acordo com o programa e comunica as decisões, através de sinais, para os dispositivos de saída.
- A fonte de alimentação é necessária para converter a tensão AC para DC (5V).
- A consola de programação é usada para inserir o programa na memória do processador e para desenvolver programas.
- A unidade de memória é o local onde o programa contém todas as ações de controlo a serem executadas pelo microprocessador e onde se armazenam os dados de entrada para processamento e para a saída.
- Os circuitos de entrada e de saída são os locais nos quais ocorre a troca de informações entre o processador e os dispositivos. O processador recebe informações externas dos dispositivos e comunica informações aos mesmos. Os sinais de entrada e de saída podem variar como discreto, digital ou analógico, como é possível observar na Figura 2.13.



(a) Discreto **(b)** Digital **(c)** Analógico

Figura 2.13: Tipos de sinais [26]

Devido à necessidade crescente do seu uso, existe uma enorme variedade de *PLC's*, baseados em processadores com vários níveis de desempenho, podendo existir, caso seja necessário, módulos para comunicações, módulos com memória adicional, entre outros [28].

2.4.3 CPU

A Unidade Central de Processamento (CPU) é responsável pela execução do programa. Dependendo do microprocessador escolhido, a estrutura interna da CPU será diferente. Assim, em geral, as CPUs têm as seguintes características:

- Possuem uma Unidade Lógica e Aritmética (ALU). Esta, é responsável pelo processamento lógico, isto é, manipulação de dados, operações aritméticas de adição e subtração e operações lógicas (*AND, OR, NOT e EXCLUSIVE-OR*).

- O seu microprocessador é composto por memória e registos que são usados para armazenar informações quando o programa está a ser executado.
- Têm uma unidade de controlo que é utilizada para controlar os tempos de operação [28].

2.4.4 Memória

Para que o PLC possa operar, é necessário que o mesmo possa aceder aos dados que vão ser processados e às instruções programadas, isto é, pode aceder ao programa que contem essas informações. Assim, ambos vão ser armazenados na memória do PLC, permitindo o acesso aos mesmos. O PLC tem vários elementos de memória. São eles:

- Memória de leitura *"Read Only Memory"* (ROM). O sistema operacional tem armazenamento permanente e dados fixos usados pelo CPU.
- Memória de acesso aleatório *"Random Access Memory"* (RAM). É usado para o programa do utilizador. A memória RAM é usada para executar dados. É o local que contem as informações relativamente aos estados dos dispositivos de entrada e saída, temporizadores, contadores e outros dispositivos internos. Esta memória é, muitas vezes, referida como uma tabela de dados ou uma tabela de registo. A memória RAM tem blocos de endereços reservados. Parte dessa memória, isto é, um bloco de endereços, será reservada para os endereços de entrada e saída e para o seu estado. Outra parte, será reservada para dados predefinidos e a última porção, para armazenar valores de contadores, temporizadores, etc.
- Memória EPROM (*Erasable and Programmable Read Only Memory*). É usada para armazenar os programas permanentemente [28].

Todos os *PLC's* têm memória *RAM* para poderem armazenar os dados e programas desenvolvidos. Esses dados e programas podem ser alterados pelo utilizador, mas, para evitar a perda de informação, quando a alimentação é desligada, é utilizada uma bateria no *PLC* para manter o conteúdo da *RAM* por um certo período de tempo. Desta forma, mais tarde, o programa pode ser carregado na memória *EPROM* (módulo aparafusado ao *PLC*) garantindo, assim, um backup do programa. Também existem armazenamentos temporários de *buffer* para entrada/saída de canais [26].

2.4.5 Unidades de Entrada e Saída (E/S)

A unidade de entrada/saída fornece a ligação física entre o sistema e o processo a controlar. Isto é, permite que sejam feitas as conexões através de canais de entrada/saída, para dispositivos de entrada,

como sensores, e dispositivos de saída, como motores. Assim, este sistema de entrada/saída, é um dos componentes mais importantes num *PLC*, pois existe uma ligação direta com o equipamento industrial, sem a necessidade de haver circuitos de adaptação. Cada entrada/saída tem um endereço único, que pode ser usado pelo *PLC*. Para facilitar a compreensão, pode imaginar-se uma rua com várias casas, em que cada casa tem um número associado. Assim sendo, cada número representa um endereço que será uma entrada ou uma saída, por exemplo o endereço '10' seria a entrada de um sensor específico e o '11' seria a saída para um determinado motor [10].

As entradas e saídas possuem um isolamento galvânico, normalmente ótico, sendo que também é isolado, eletricamente, do ambiente externo. Desta forma, podem estar em zonas de elevado ruído elétrico, pois este isolamento dá uma melhor fiabilidade e segurança na comunicação com os sensores e atuadores. Este isolamento também permite que uma ampla gama de sinais de entrada, seja fornecida ao *PLC*, isto é, existem diferentes níveis de tensão e potência entre o *PLC* e os processos a controlar. As saídas são especificadas como sendo do tipo relé, transístor ou triac, podendo, também, ter uma ampla gama de sinais caso necessário [28].

2.4.6 Terminal de programação

Antes de existirem computadores pessoais, a programação dos *PLC's* era feita através de terminais de programação dedicados. Estes terminais eram equipamentos robustos, que tinham teclado e monitor, como se pode observar na Figura 2.14.

Figura 2.14: Terminal de programação [28]

Com o desenvolvimento tecnológico, foram criados terminais que eram possíveis de programar *offline*, isto é, sem se estar ligado ao *PLC*. Era possível desenvolver o programa num local distante do *PLC*, sendo preciso apenas regressar para efetuar pequenas mudanças necessárias.

Os protocolos usados na comunicação entre os terminais de programação eram desenvolvidos pelos fabricantes de PLCs, o que não permitia a utilização do mesmo terminal de programação em diferentes marcas de PLCs. As linguagens de programação, como as teclas de atalho nos terminais, eram diferentes de acordo com a marca, tendo, desta forma, funcionalidade reduzida, pois apenas eram compatíveis com a marca que os produziu, sendo apenas utilizados para alterar pequenos códigos ou alterar dados. Havia outros dispositivos mais baratos, como unidades de fita perfurada e fita magnética, para armazenar os programas desenvolvidos, sendo as fitas perfuradas mais baratas e resistentes em meio ambiente fabril [28].

2.4.7 Norma IEC 61131

Em 1993, a IEC publicou a primeira edição da norma IEC 61131, com objetivo de estabelecer padrões para os PLCs [29].

Esta norma foi dividida em vários capítulos. São eles:

- 61131-1 - Definição geral e conceitos básicos.
- 61131-2 - Requisitos de hardware.
- 61131-3 - Linguagens de programação.
- 61131-4 - Guia sobre seleção, instalação e manutenção.
- 61131-5 - Comunicação com outros dispositivos.
- 61131-6 - Comunicação ("*Communications via field bus software facilities*").
- 61131-7 - Programação de controlo "*fuzzy*".
- 61131-8 - Guia para a implementação de linguagens de programação PLC definida na parte 3 [26].

2.4.8 Linguagens de programação para PLC

A programação de PLCs utiliza software a partir de um computador pessoal (PC) ou pelas consolas fornecidas pelo construtor. Para ajudar os engenheiros com capacidades reduzidas de programação, foi desenvolvida a programação *ladder*. Grande parte dos fabricantes escolheu esse método, mas desenvolveu as suas próprias versões. Por causa disso, em 1993, a *International Electrotechnical Commission* (IEC) publicou a norma 1131-3, conhecida como a norma IEC 61131-3.

As linguagens de programação são divididas em dois grupos distintos, nomeadamente, Linguagens Gráficas e Linguagens Textuais. As linguagens gráficas fornecem um conjunto de símbolos gráficos que são ligados entre si, constituindo um programa. A linguagem textual é uma linguagem cujas instruções

são escritas em texto, usando palavras reservadas dessa linguagem. Nas linguagens gráficas podemos destacar a linguagem LD - *Ladder Diagram*, ou diagrama *Ladder* e FBD - *Function Block Diagram*, ou diagrama de blocos de funções. Na linguagem textual podemos destacar a linguagem IL - *Instruction List*, ou lista de instruções e ST - *Structured Text*, ou texto estruturado. Existe outra linguagem de programação, SFC – *Sequential Function Chart* ou gráfico de função sequencial, que possui elementos para organizar o programa de maneira sequencial e permite o controlo paralelo de processos [26].

2.4.9 Linguagem *Ladder*

Como foi referido anteriormente, *ladder* ou escada foi uma das primeiras linguagens a ser utilizada na programação de PLC. Esta linguagem permitiu a técnicos e engenheiros da área de automação, que possuíam conhecimentos da lógica tradicional de contactos, mas pouco de programação, pudessem programar PLCs [30].

A linguagem *ladder* consiste numa estrutura de linhas e colunas, existindo em cada linha várias instruções (contactos) e um resultado (saída). Várias linhas possuem a aparência de degraus, o que leva ao termo escada (*ladder*), representada na Figura 2.15.

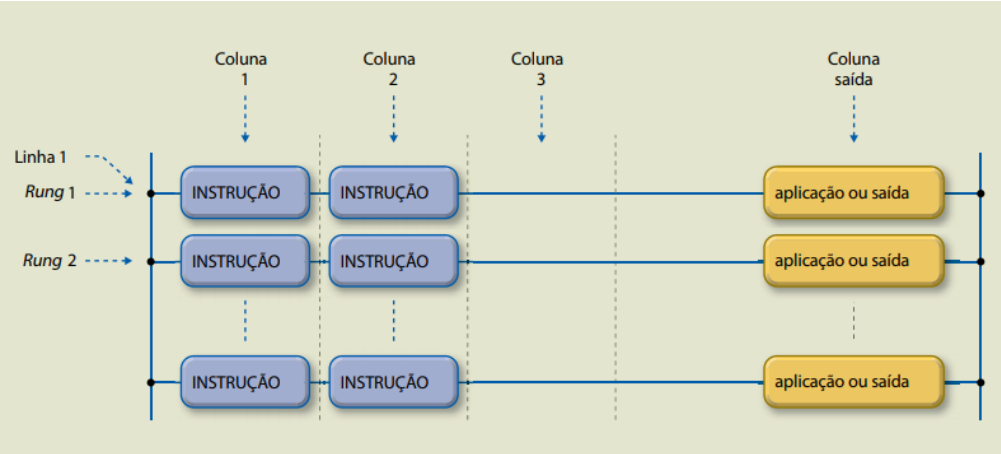


Figura 2.15: Terminal de programação [28]

Esta linguagem é composta por 6 categorias de instruções, nomeadamente, do tipo relé, de temporização e contagem, de manipulação de dados, aritméticas, de transferência de dados e de controlo de programa.

A Figura 2.16 mostra a estrutura básica de um degrau. A saída só irá ser atuada caso algum caminho esteja fechado, desde o início até ao fim, isto é, quando existir continuidade lógica.

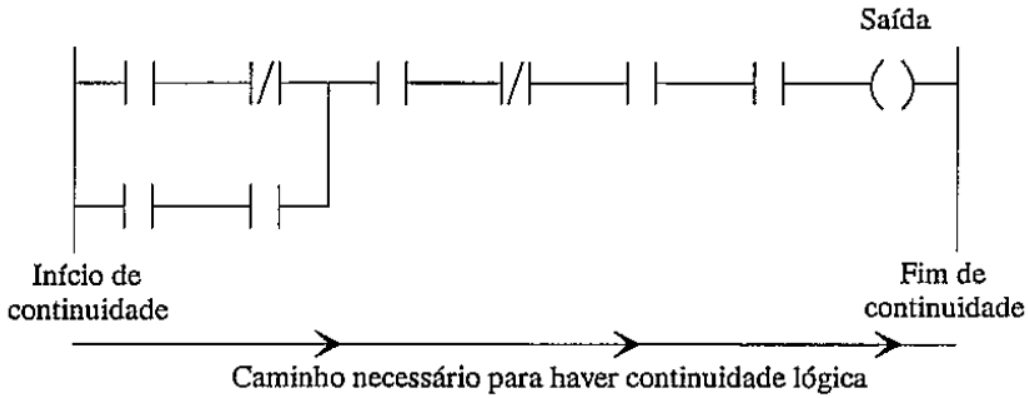


Figura 2.16: Degrau num diagrama de escada [27]

Nesta linguagem, as saídas, chamadas de bobinas, e os contactos são os símbolos básicos da lista de instruções. Como se pode verificar através da observação da Figura 2.16, a posição dos contactos nos degraus depende da lógica que se quer aplicar. Assim, podem ter-se contactos dispostos em série, paralelo ou ambos, dependendo do controlo desejado. A sua programação consiste na atribuição de endereços que identificam o que está a ser controlado, verificando se está ativo ou não. Cada endereço identifica uma saída ou uma entrada, referenciando a localização de um ponto interno da memória [27].

Para consolidar o tipo de linguagem utilizado, foi realizada uma análise das principais instruções. Tal, está apresentado no apêndice A.

2.4.10 *Human Machine Interface*- HMI

Traduzindo do inglês *Human Machine Interface*, Interface Homem Máquina, podemos observar que esta consola cria uma ligação entre o ser humano e as máquinas. Consolas HMI são ecrãs de cristal líquido (*Liquid Crystal Display* (LCD)) em que é possível programar uma interface entre o utilizador e o sistema. Neste caso, a consola está interligada com o controlador, podendo alterar estados das variáveis através da consola e enviando sinais (mensagens) para o PLC.

O principal objetivo das consolas HMI consiste em substituir os botões e atuações em relés do sistema. Deste modo, consegue-se verificar e alterar o estado do sistema, isto é, observar o estado das entradas e de saídas, como por exemplo, poderemos observar se um sensor está ativo ou não, atuar um motor, etc. É, ainda, possível criarem-se mensagens de alerta e de emergência com indicação do tempo e localização, efetuar a leitura de dados do sistema (número de produção de peças e realização de gráficos com os dados), permite ao operador alterar temporizadores (*timer*) e contadores apenas tocando no ecrã, etc.

Assim sendo, as aplicações desenvolvidas nas consolas HMI têm um papel fundamental na gestão das operações do sistema, pois determinam a facilidade que o operador irá encontrar ao interagir com a consola. As principais tarefas no desenvolvimento de uma aplicação para uma consola HMI são: [34]

- A comunicação com o PLC;
- A criação das variáveis que estão endereçadas no PLC;
- A inserção de gráficos visíveis na consola;
- Os dois tipos básicos de animações de objetos: o primeiro permite que os parâmetros sejam exibidos; o segundo é usado para a intervenção do utilizador e permite ao mesmo alterar os parâmetros do sistema.

Existem muitos tipos diferentes de consolas HMI, com diferentes características a nível de software e hardware. As resoluções dos ecrãs das consolas são expressadas por comprimento e altura e utilizam como unidade o número de pixeis. Os ecrãs podem ser tácteis e podem conter cores, sendo necessário seguir uma convenção para a utilização da mesma: [34]

- Vermelho - Alarme, perigo e parar;
- Amarelo - Precaução e risco de perigo;
- Verde – Condições prontas, em funcionamento e seguras.

Através da consola é possível observarem-se informações sobre eventos e alarmes, contendo a sua data de ocorrência e duração. Estas informações podem ser armazenadas em tabelas, de forma a que o utilizador possa aceder quando necessário.

Capítulo 3

Trabalho Desenvolvido

Neste capítulo é debatido o problema atual e demonstrado o trabalho desenvolvido. Assim, é analisada a solução existente bem como os problemas encontrados na mesma. É, também, apresentado um diagrama que explica a unidade de controlo, bem como as ligações dos componentes da célula. Explica-se a comunicação entre esses componentes e as suas ligações físicas. Tentaram encontrar-se soluções de baixo custo, quer para os problemas descobertos, quer para não aumentar o orçamento já despendido na célula. Após isto, implementaram-se as soluções encontradas e programaram-se os robôs, o *PLC* e o *HMI*. Por fim, simulou-se o funcionamento da célula, otimizando-a o máximo possível.

3.1 Análise da primeira solução encontrada

Após efetuado o estudo sobre os fundamentos teóricos, analisou-se a solução mais recente. Visto que é um projeto começado em 2015, foi sofrendo alterações e melhorias, mas nunca foi possível produzir grandes quantidades de quadros com um erro admissível.

A solução encontrada prevê a utilização de dois robôs *KUKA* (Kr 16-2), de uma mesa de indexação (para montagem do painel com as peças de plástico e metal), de uma mesa de fixação (para o aparafusamento dos quadros montados), de um transportador de entrada de acessórios (transporta as peças de plástico e metal), de uma pilha de entrada de painéis (armazém de painéis) e, por fim, de um transportador de saída (transporta o quadro completo para a célula seguinte).

O funcionamento da célula envolve a utilização do robô 1 para pegar num painel branco e nas restantes peças (os perfis de alumínio e as peças plásticas). Este robô manipula individualmente cada peça colocando-as na mesa de indexação. Em primeiro lugar, coloca o painel branco, em segundo, coloca as restantes peças, ocorrendo assim a montagem do quadro. De seguida, o robô 2 pega no quadro já montado e coloca-o na mesa de fixação onde irá ocorrer o aparafusamento. Por fim, o quadro completo avança para um tapete terminando assim o processo nesta célula.

3.1.1 *Layout* da primeira solução

Para uma observação mais atenta da primeira solução, encontra-se na Figura 3.1, o posicionamento dos vários componentes referidos anteriormente. Assim, pode ser observado que, para além desses componentes, ainda é visível uma rede de proteção, bem como duas portas de acesso à célula e dois controladores correspondentes a cada robô.

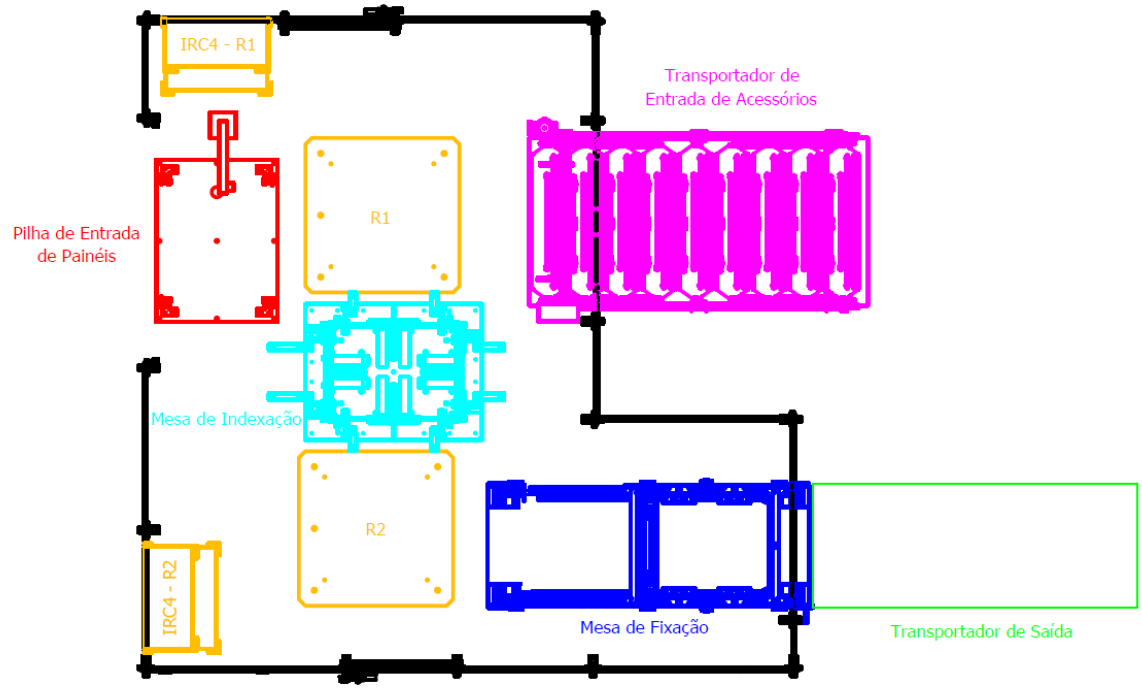


Figura 3.1: Layout da primeira solução

3.1.2 Quadros completos

O objetivo principal desta célula é a produção de quadros. Desta forma, para entender totalmente o objetivo proposto deve observar-se atentamente a Figura 3.2. Esta realça um quadro completo, totalmente montado e aparafusado. O mesmo é constituído por duas peças de plástico, duas peças de metal e um painel branco. Observa-se, também, que as peças de metal são iguais enquanto que as peças de plástico são diferentes, portanto é necessário ter atenção na mesa de indexação (não trocar a peça de topo pela peça de base).

Figura 3.2: Quadro Completo

3.1.3 Verificação dos esquemas e do material existente

Este projeto foi sofrendo alterações ao longo do tempo, visto que foi iniciado em 2015 e passando por diferentes pessoas. Desta maneira, e antes de ser possível implementar uma solução ou fazer testes à célula, foi necessário verificar o esquema elétrico e pneumático, assim como o material existente na empresa *RobotSol*.

O esquema elétrico estava muito desatualizado, então, foi decidido pela *RobotSol* que seria atualizado no fim do projeto. Isto é, quando se fosse implementar no cliente iria atualizar-se o esquema.

O esquema pneumático encontra-se no apêndice B. O mesmo estava desatualizado em relação à realidade, isto é, existiram melhorias que aconteceram na realidade mas que nunca chegaram a ser atualizadas no esquema e vice versa. Devido a isso, foi feita uma comparação entre a realidade e o esquema, atualizando-se tanto o esquema pneumático como os componentes necessários na célula.

Por fim, como se pode confirmar pelo apêndice C, fez-se um levantamento do material da célula. Algum do material não se encontrava na empresa e outro não estava montado, sendo necessário realizar-se esse procedimento.

3.1.4 Implementação da primeira solução

Ao observar a Figura 3.1 foi possível posicionar os componentes nos seus devidos locais, isto é, recriou-se a célula como seria no cliente.

A Figura 3.3 retrata os vários componentes colocados na empresa *RobotSol*. A Figura 3.3a e a Figura 3.3b apresentam a mesa de indexação, o transportador de entrada de acessórios, o robô 1 e uma pilha de entrada de painéis (armazém). A Figura 3.3c e a Figura 3.3d expõem a mesa de fixação e o robô 2. Por fim, ainda é possível observar, na Figura 3.3b e na Figura 3.3d, uma linha branca no chão que representa o limite da célula, isto é, a rede de proteção.

Com esta implementação foi possível testar os componentes como se estes estivessem no cliente, ajudando a solucionar os problemas e a otimizar a célula.

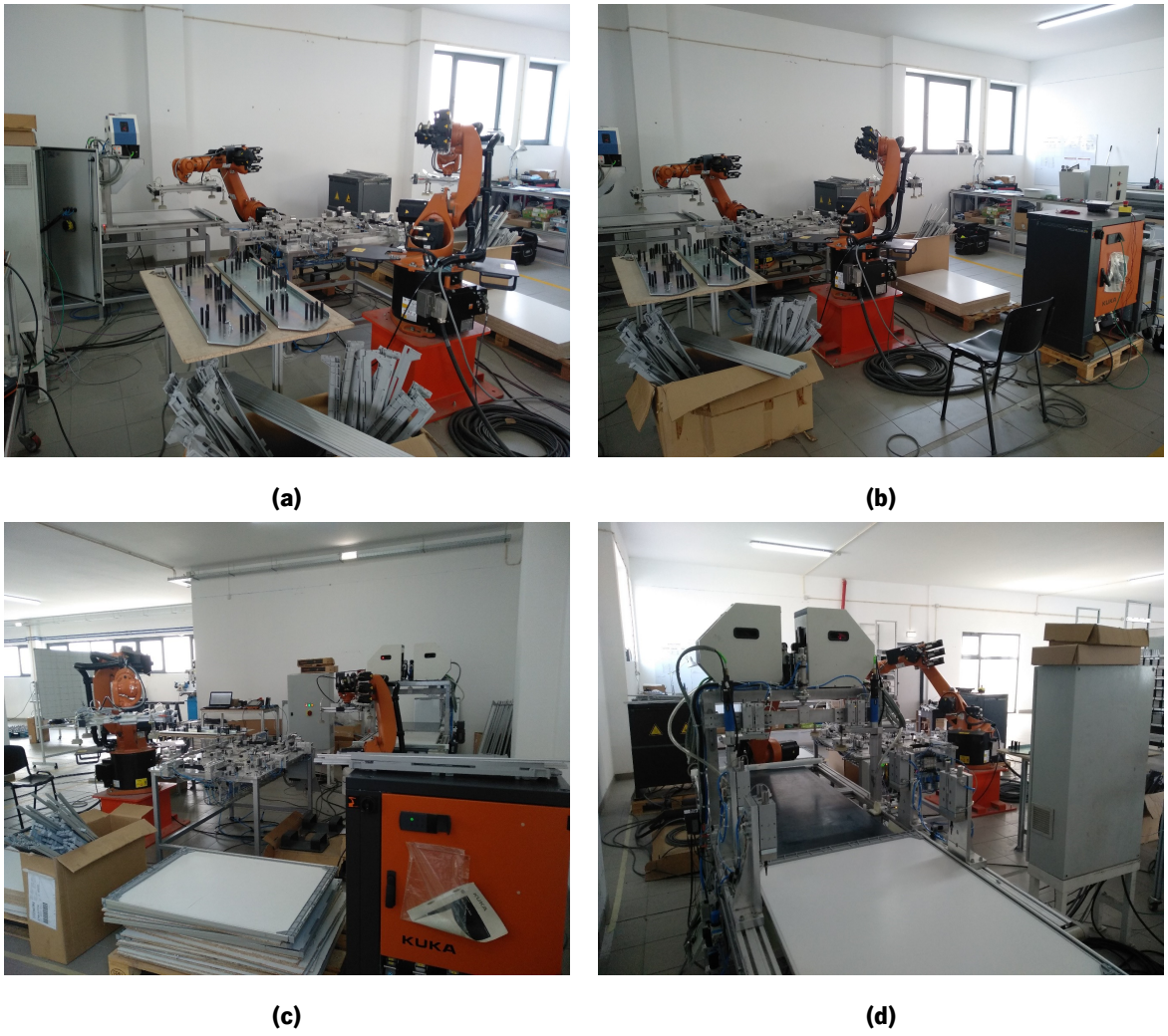


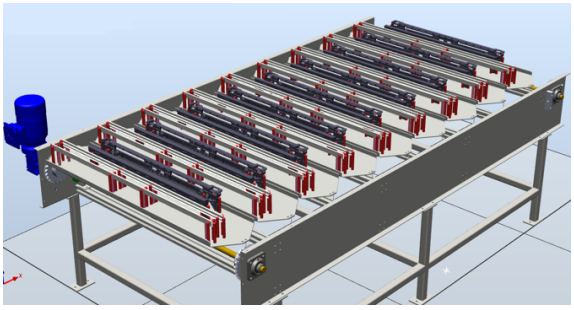
Figura 3.3: Implementação da primeira solução na *RobotSol*

3.1.5 Transportador de entrada

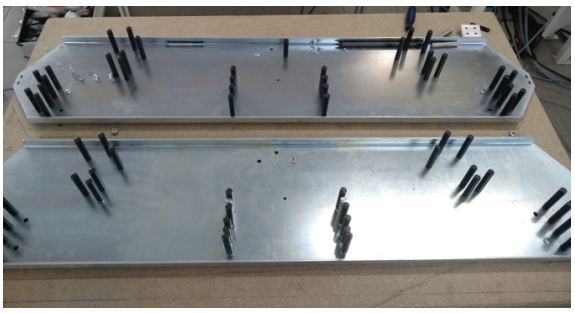
O transportador de entrada é o local onde foram colocadas todas as peças metálicas e plásticas. A Figura 3.4 evidencia duas imagens, uma correspondente à simulação e a outra à implementação na empresa.

A primeira Figura 3.4a mostra o funcionamento do transportador de entrada, um tapete rolante com várias peças metálicas, em que cada peça metálica é constituída pelas peças de plástico e metal. O transportador de entrada vai avançando as peças à medida que o robô retira as quatro peças, isto é, só avança um novo conjunto quando o primeiro estiver vazio.

A segunda Figura 3.4b retrata a implementação na empresa *Robotsol*. Pode observar-se, que cada peça tem uma posição fixa e que se encontra na mesma coordenada, e, desta forma, o robô tem as mesmas coordenadas para pegar nas peças. Devido ao facto de o transportador de entrada se encontrar no cliente, foram utilizadas duas peças metálicas para simular a realidade, obtendo-se assim o mesmo resultado do que o transportador de entrada completo.



(a) Esquemático 3D do transportador de entrada



(b) Transportador de entrada vazio

Figura 3.4: Transportador de entrada

3.1.6 Transportador de saída

O objetivo do transportador de saída, como o nome indica, transporta o quadro completo para a célula seguinte, isto é, faz a ligação entre células. O seu funcionamento envolve um motor, que faz rodar o tapete, deslocando o quadro completo até o mesmo ser detetado por um sensor. Quando o sensor é ativado (deteta um quadro), o motor para e o quadro encontra-se na coordenada desejada para ser utilizado noutra célula. Devido à mesa de fixação ser constituída por um motor e um sensor igual ao transportador de saída, estando o mesmo no cliente, não houve necessidade de o utilizar na *Robotsol*.

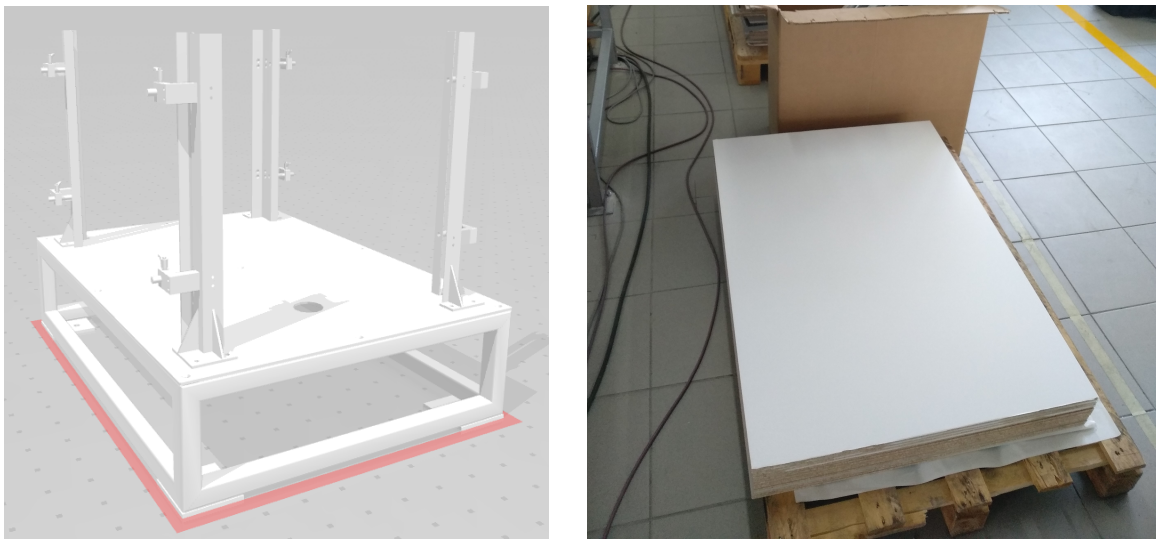
3.1.7 Alimentador de placas

O alimentador de placas é o local onde se colocam todos os painéis brancos. Pela observação da Figura 3.5 verificam-se duas imagens, uma correspondente à imagem simulada e a outra à implementação na empresa.

Pela Figura 3.5a, é possível verificar que existe uma estrutura para orientar as placas, posicionando-as

sempre na mesma posição (robô pega na placa sempre no mesmo ponto modificando apenas a altura do mesmo) e verifica-se a ausência do sensor ótico de nível, *vide* apêndice C, lista de materiais. O sensor ótico mede a altura a que se encontram os painéis brancos. Este sensor tem uma saída analógica que indica a distância do sensor à placa, sendo, deste modo, possível calcular a altura que se encontra a última placa, bastando apenas saber a altura que o sensor se encontra e a altura mínima que o painel pode estar. Desta forma, sabe-se qual a distância (eixo Z) necessária para o ponto marcado com o robô 1 para o mesmo conseguir pegar nas placas.

Pela Figura 3.5b verifica-se que apenas foram colocados os painéis em cima de uma palete, simulando, assim, a estrutura. Por fim, observa-se uma discrepância entre a simulação e a realidade, devido ao material descrito não se encontrar na *RobotSol* mas sim no cliente.



(a) Esquemático 3D do alimentador de placas

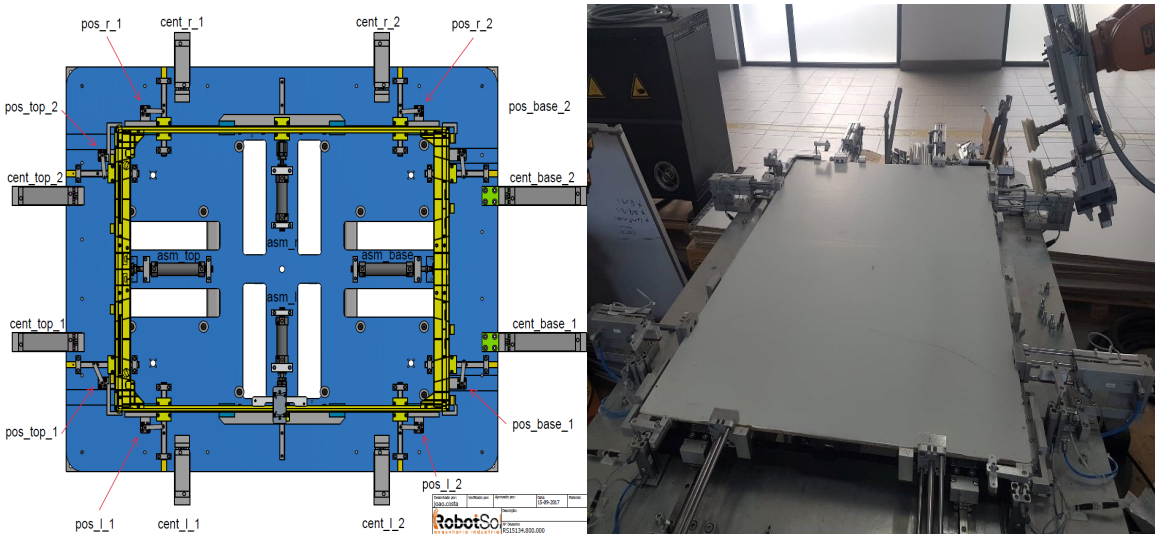
(b) Alimentador de placas real

Figura 3.5: Alimentador de placas

3.1.8 Mesa de indexação

Como já foi referido anteriormente, é na mesa de indexação que se encaixa o painel branco com as restantes peças (metal e plástico). Recorre-se à Figura 3.6 para explicar o funcionamento da mesa de indexação. Assim, o seu funcionamento começa com o robô 1 a colocar a primeira peça (painel branco) no centro da mesa de indexação. De seguida, observa-se na Figura 3.6b que os cilindros pneumáticos são utilizados para garantir que o painel se encontra numa posição central. De seguida, utiliza-se uma ventosa (que se encontra no centro da mesa) para criar vácuo. Este método é utilizado para não se perder a posição do quadro, ficando o mesmo sempre na mesma coordenada. O robô 1 coloca individualmente as duas peças de metal nas laterais da mesa. Observam-se na Figura 3.6a quatro cilindros pneumáticos

(*pos_r_1, pos_r_2, cent_r_1, cent_r_2*) , os dois primeiros têm a função de fixar as peças metálicas, enquanto que os segundos empurram as peças metálicas para encaixar no painel. É necessário fixar as respetivas peças para que as mesmas não caiam ou percam a posição desejada. Deste modo, são encaixadas as peças metálicas no painel. Por fim, para a anexação das duas últimas peças plásticas, utiliza-se o mesmo procedimento citado. O robô 1 coloca as peças nas posições laterais, os cilindros pneumáticos fixam-nas (para elas não perderem a sua posição) e, por fim, dois outros cilindros pneumáticos empurram-nas contra o painel encaixando-as e criando assim um quadro completo.



(a) Esquemático 3D da mesa de indexação **(b)** Mesa indexação real

Figura 3.6: Mesa indexação

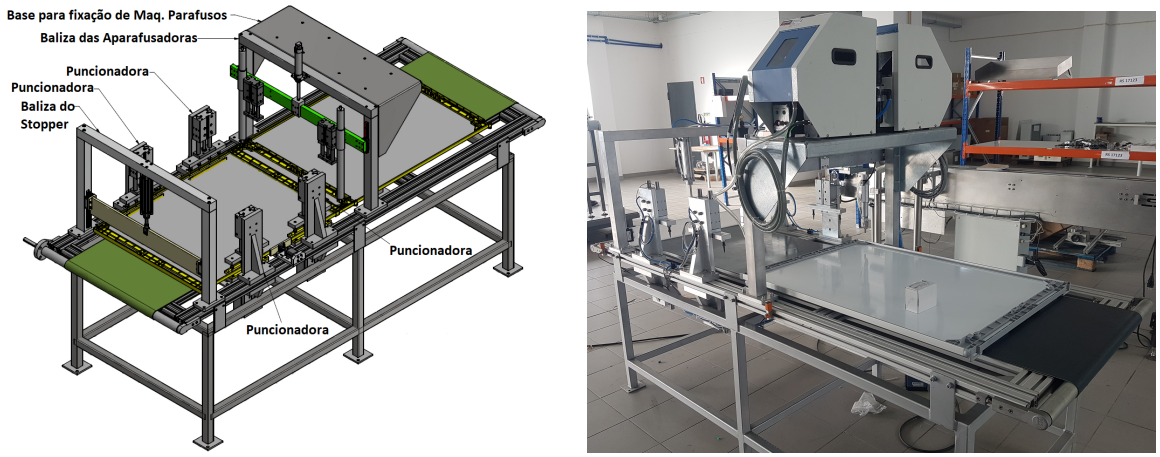
3.1.9 Mesa de fixação

A mesa de fixação é o local onde se aparafusa e deforma o quadro já montado. Existem quatro pontos diferentes situados nos quatro cantos do quadro, locais onde este é aparafusado. Assim, nestes locais, ocorre a interceção entre as peças de plástico, as peças de metal e o painel. É necessário aparafusar o quadro para que o mesmo não se desmonte aquando da utilização. O local onde se amolga o quadro situa-se nas laterais, diretamente nas peças de metal, isto é, dobram-se as peças, colocadas anteriormente, para o painel não deslizar e ficar fixo ao quadro.

A Figura 3.7 pretende explicar o funcionamento da mesa de fixação. O seu funcionamento começa com o robô 2 a colocar o quadro completo na mesa de fixação. Observa-se na Figura 3.7b que o quadro encontra-se na posição de aparafusamento, ocorrendo o processo de aparafusar. Para tal, em primeiro lugar, é necessário avançar um cilindro pneumático que baixa as duas aparafusadoras para a posição correta. Em segundo lugar, avançam os cilindros pneumáticos, que se encontram debaixo das aparafusa-

doras, suportando a força que a mesma executa. Por fim, é necessário avançar dois cilindros pneumáticos para ocorrer o aparafusamento. De seguida, é necessário recuar os cinco cilindros e ativar um motor que avance o tapete, para gerar o movimento do quadro até o mesmo colidir no *stopper* final. Na Figura 3.7a, pode-se ver o *stopper* que se encontra à esquerda da mesa. O mesmo tem o cilindro pneumático avançado para poder parar o quadro. Mal o quadro toque no *stopper*, o mesmo encontra-se na segunda posição de aparafusamento. Ocorre, novamente, o aparafusamento do quadro, como anteriormente explicado e, após isso, dá-se o procedimento de amolgar o mesmo. É necessário que os oito cilindros pneumáticos (quatro de topo e quatro de base) avancem. Os cilindros de topo, com duas ponteiras cada, avançam contra o metal e amolgam-no. Os cilindros de base suportam a força exercida. Por fim, recuam-se todos os cilindros pneumáticos (procedimento de amolgar e do *stopper* final) e o quadro avança para a célula seguinte.

Explicado o funcionamento da mesa, incide-se noutro componente importante, o alimentador de parafusos. Este componente, como o nome indica, alimenta a aparafusadora associada ao mesmo. É possível observar na Figura 3.7b os dois alimentadores de parafusos por cima da mesa de fixação, cada um para cada aparafusadora. Este componente, para além de fornecer os parafusos, também os armazena. O seu funcionamento é simples, ou seja, após o aparafusamento, o componente utiliza o ar para enviar um parafuso, na posição correta, para a aparafusadora, repetindo-se o processo.



(a) Esquemático 3D da mesa de fixação

(b) Mesa de fixação real

Figura 3.7: Mesa de fixação

3.1.10 Controladores, robô 1 e robô 2

Como referido anteriormente, nesta célula são utilizados dois controladores KR C3, Figura 3.8a e dois robôs Kr 16-2, Figura 3.8b. Tendo a empresa *RobotSol* o equipamento citado, foi necessário fazer as

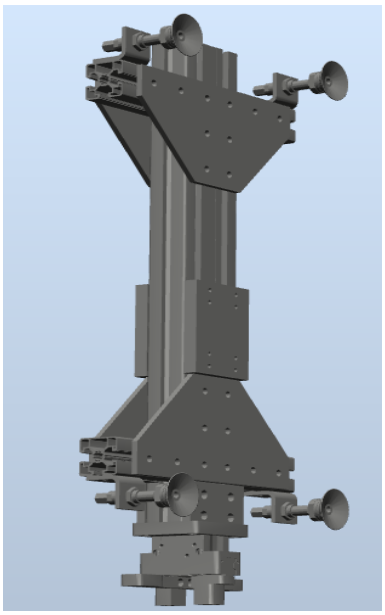
ligações físicas dos mesmos. Estas ligações envolvem o cabo de alimentação, o cabo de dados, o cabo de ligação para a consola *KUKA smartPAD* e o cabo de segurança física (x11). O cabo 'x11' é um cabo de segurança física que impede o robô de operar à velocidade máxima, sendo que, quando o mesmo não está ligado, o robô só opera no modo T1. Como só existia um cabo 'x11', só foi possível testar um robô de cada vez à velocidade máxima.

(a) Controlador KR C3 **(b)** Robô KR 16-2

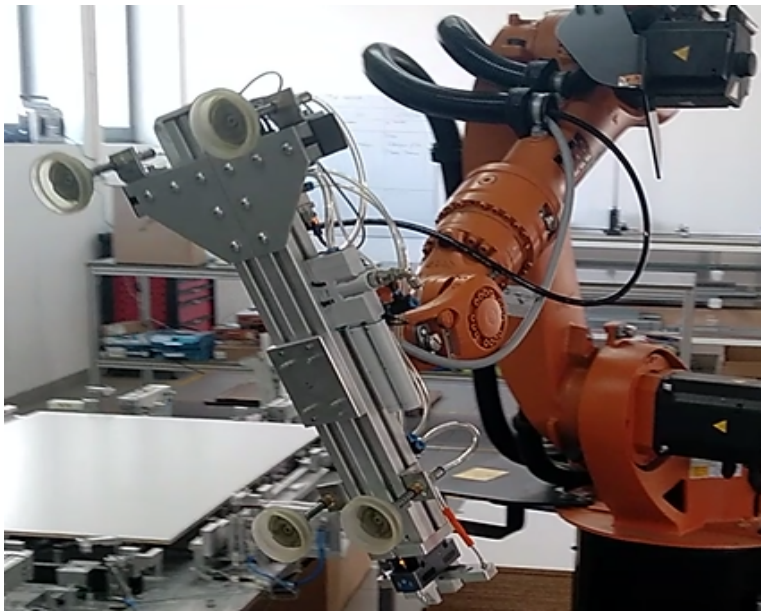
Figura 3.8: Controlador e robô

3.1.11 *Gripper* robô 1

Na Figura 3.9, verifica-se que a ferramenta utilizada pelo robô 1 apresenta dois componentes. O primeiro componente é composto por quatro ventosas e o segundo por uma garra. Cada componente tem uma função diferente, sendo os dois componentes pneumáticos. As ventosas em contacto com o painel branco criam uma força de sucção capaz de suportar o peso do mesmo, servindo assim para o transportar para os locais desejados. A garra tem o papel de agarrar, individualmente, em cada peça de plástico ou metal. Na Figura 3.9b é possível observar duas electroválvulas (biestável e monoestável) e um gerador de vácuo.



(a) Esquemático 3D *gripper* robô 1

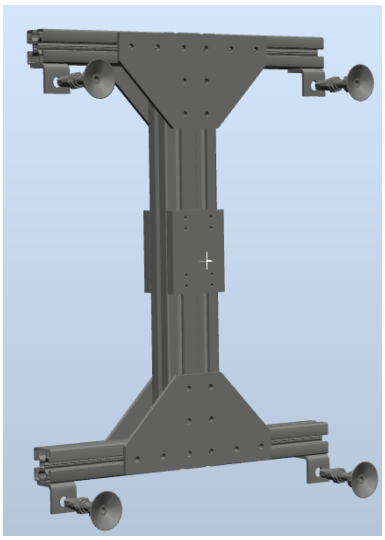


(b) Gripper robô 1 real

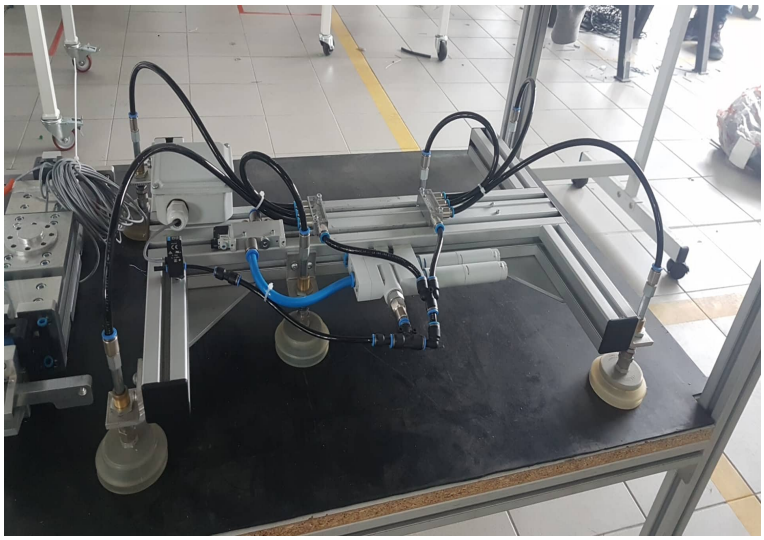
Figura 3.9: *Gripper* robô 1

3.1.12 *Gripper* robô 2

Na Figura 3.10, verifica-se que a ferramenta utilizada pelo robô 2 apresenta apenas um componente constituído por quatro ventosas. Estas têm as mesmas caraterísticas que as explicadas anteriormente. Na Figura 3.10b é possível observar uma válvula solenoide simples e um gerador de vácuo.



(a) Esquemático 3D *gripper* robô 2



(b) *Gripper* robô 2 real

Figura 3.10: *Gripper* robô 2

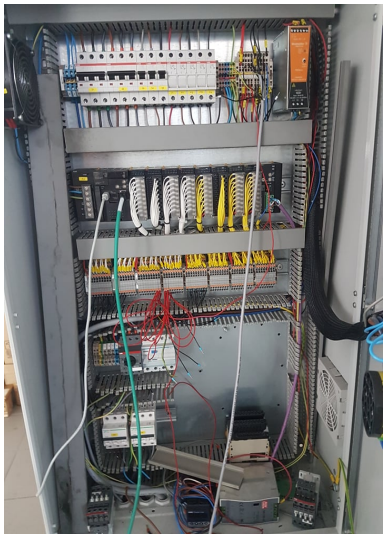
3.1.13 Quadro elétrico

O quadro elétrico, ou quadro de distribuição, é um equipamento destinado a receber energia elétrica de uma ou mais fontes de alimentação e distribuí-las a um ou mais circuitos. Tem um conjunto de diversos aparelhos de proteção e diversos componentes dependentes da função para que foram concebidos [35].

Na Figura 3.11a, observa-se o quadro implementado no cliente. O mesmo é constituído por uma consola *HMI* e vários botões. No seu interior, encontra-se o *PLC - OMRON CJ2M-CPU32* bem como as suas entradas e saídas, equipamentos de proteção, barramentos e outros componentes necessários ao funcionamento da célula. Verifica-se na zona da botoneira, situada debaixo do *HMI*, que a mesma é formada por três botões (*start*- verde, *stop*- vermelho e *rearme*-laranja) e uma botoneira de paragem de emergência. Por fim, é visível, na zona por cima do *HMI*, três sinalizadores correspondentes a cada fase e uma botoneira de corte geral de energia.

Pelo facto do quadro elétrico não se encontrar na *Robotsol*, este foi substituído pelo quadro de testes, representado na Figura 3.11b.

(a) Quadro elétrico implementado no cliente



(b) Quadro elétrico para testes

Figura 3.11: Quadro Elétrico

3.1.14 PLC *OMRON CJ2M-CPU32*

O controlador lógico programável (*PLC*) utilizado nesta célula é da marca *OMRON* e o modelo é *CJ2M-CPU32*. O *PLC* é responsável pelo controlo do sistema, comunicando através do protocolo de comunicação *DeviceNet* e possui um papel crucial neste sistema, visto que consiste na gestão de espaço da célula. Isto é, ambos os robôs interagem com a mesa de indexação, logo é necessária a gestão dos

mesmos de forma a impedir colisões. Também o *PLC* é responsável por controlar a sequência na mesa de indexação e fixação. Tudo isto é realizado através de trocas de sinais que o *PLC* realiza com as mesas e robôs.

O *PLC* é responsável pelas seguintes funções na célula:

- Arranque em automático de todos os componentes da célula;
- Gestão de segurança do sistema, controlo de botoneiras ou abertura de portas;
- Comunicação entre robôs;
- Atuação das eletroválvulas tanto a nível das ferramentas dos robôs, como a nível das mesas;
- Envio e receção de informação proveniente da consola *HMI*.

Figura 3.12: *PLC OMRON CJ2M-CPU32*

3.1.15 Consola *HMI OMRON nb10w-tw01b*

Como referido anteriormente, a consola *HMI* é o equipamento responsável pela interação do utilizador com a célula. Esta consola tem um tamanho de 210.8 mm × 268.8 mm × 54.0 mm e o seu *datasheet* encontra-se no Anexo C. Esta interface foi programada, especificamente, para o sistema em questão, sendo possível verificar dados e estatísticas de produção, como por exemplo, o número total de peças produzidas. Também se visualiza o estado do sistema e alarmes. Na consola, realiza-se uma gestão de utilizadores (adicionar ou remover) e configuram-se níveis de permissão.

3.2 Diagrama do sistema

Nesta secção, apresenta-se o diagrama de controlo do sistema. Assim, demonstra-se a relação do *PLC* e do controlador de cada robô com os vários sinais de entrada e saída, interfaces e comunicações. Após esta explicação, apresenta-se a arquitetura geral de comunicações do sistema.

No funcionamento geral do sistema existe a interligação entre os diferentes componentes. Na Figura 3.13 é apresentado o diagrama de controlo relativamente aos controladores do sistema. Pela figura observam-se quatro grupos. O grupo central (Unidade de Controlo) é o núcleo desta parte do sistema. Aqui encontram-se os dois controladores dos robôs, *KUKA KR-C4*, e o *PLC* utilizado, *OMRON CJ2M-CPU32*. Pode afirmar-se que o *PLC* controla todos os acontecimentos da célula, citando-se como exemplos: arranque dos robôs; controlo de segurança do sistema; gestão de áreas ocupadas; gestão da montagem do quadro (informa o robô 1 quando pode colocar as peças na mesa de indexação, informa o robô 2 quando pode ir buscar o quadro montado, gestão da sequência de fixação, entre outros). O controlador do robô realiza a lógica de funcionamento do mesmo, ou seja, verifica as condições necessárias para o robô realizar os movimentos adequados. Pode concluir-se que o robô é o *slave* e o *PLC* é o *master*, pois o robô só se poderá movimentar quando o *PLC* permitir, pois este é sempre o que dá a ordem final.

O grupo "Entradas"consiste nas informações que as unidades de controlo adquirem dos diferentes componentes existentes no sistema. Essas informações, retiradas dos sensores do sistema, são sinais que representam a posição dos cilindros pneumáticos e dos sensores da mesa de indexação e fixação, que detetam objetos (por exemplo, uma peça ou o quadro completo) e que representam o estado das botoneiras de emergência e da porta de segurança.

No lado oposto encontra-se o grupo "Saídas"que representa as ações que a unidade de controlo executa no sistema. Neste grupo são atuados equipamentos como os cilindros pneumáticos, através de sinais digitais, é gerida a segurança do sistema, podendo permitir ou não o acesso à célula e é dada a autorização dos movimentos do robô, por exemplo: o robô 1 pode colocar uma peça na mesa de indexação, ou ser proibido de movimentos se a área já estiver ocupada; o robô 2 pode estar livre para pegar no quadro completo mas o robô 1 ainda está na área envolvida, então o robô 2 espera pela ordem do *PLC* que informe que a área esteja desocupada.

Por último, no grupo "Interfaces e Comunicações"existe uma troca de informação com a unidade de controlo. Por exemplo, a comunicação presente é a troca de informação utilizando o protocolo *DeviceNet*. A nível de interfaces, existe a consola de programação do robô e a consola *HMI*, já explicadas anteriormente. Na primeira, é possível visualizar o estado dos sensores presentes no *gripper*, bem como alterar e visualizar o valor das variáveis de entrada ou saída do robô. Também se podem observar os alarmes e erros que existem no robô. Utilizando a consola *HMI* pode-se atuar qualquer cilindro pneumático desejado, observar ou alterar as variáveis de entrada ou saída, visualizar erros ou alarmes, entre outras funções.

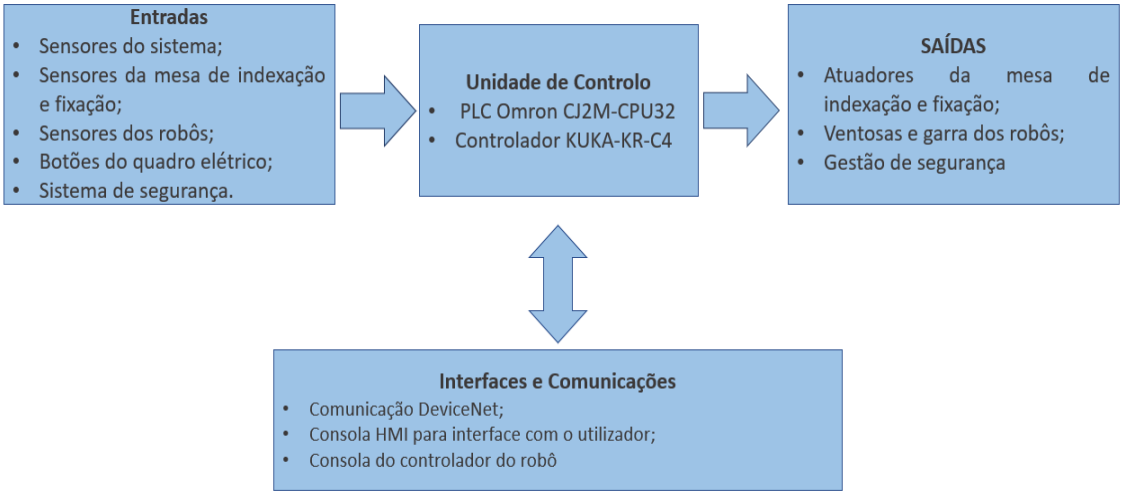


Figura 3.13: Diagrama de controlo

Na Figura 3.14 são apresentadas todas as ligações que os componentes têm entre si. Como se verifica, para ativar um cilindro pneumático na mesa de fixação através da consola *HMI*, é necessário a mesma comunicar com o *PLC*, que por sua vez comunica com a mesa de fixação. Este é um exemplo de como se pode comunicar entre componentes que não estão diretamente ligados.

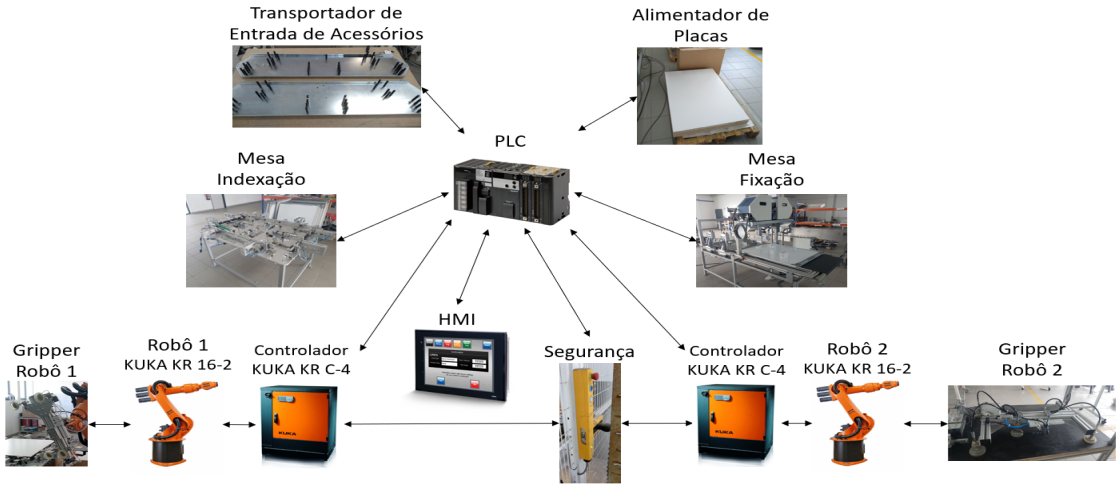


Figura 3.14: Arquitetura geral de comunicações do sistema

3.3 Comunicação *DeviceNet*

Para existir comunicação entre os vários componentes é necessária a realização de ligações físicas entre os mesmos. Posteriormente, foi criada a rede e foram feitas as configurações necessárias, tanto no *PLC*, como nos robôs *KUKA*.

Assim, são necessárias três ferramentas de trabalho:

- *WorkVisual* 5.0 (configurar os robôs).
- *Cx-Programmer* 9.70 (configurar o *PLC*).
- *Cx-Integrater* 2.66 (configurar a rede).

O procedimento de implementação da comunicação *DeviceNet* aqui representado, é constituído pela seguinte sequência de módulos:

- CJ1W-DRM21 (Módulo do *PLC*) > BK5250 (Módulo na mesa de indexação) > KR C 4 (R1) > KR C 4 (R2)

Os componentes constituintes para a comunicação desejada possuem as seguintes unidades:

Mesa de Indexação

- 1 unidade de BK5250: Unidade de cabeceira para *DeviceNet*;
- 3 unidades de KL1809: Entradas digitais;
- 1 unidade de KL2809: Saídas digitais;
- 1 unidade de KL3052: Entradas analógicas;
- 1 unidade de KL9010: Terminal de extremidade do barramento.

Robô 1

- 1 unidade de EK1100: *EtherCAT*;
- 1 unidade de EL6752: *DeviceNet*;
- 1 unidade de EL9100: Terminal de fonte de alimentação;
- 1 unidade de EL1809: Entradas digitais;
- 1 unidade de EL2809: Saídas digitais.

Robô 2

- 1 unidade de EK1100: *EtherCAT*;
- 1 unidade de EL6752: *DeviceNet*;
- 1 unidade de EL9100: Terminal de fonte de alimentação;
- 1 unidade de EL1809: Entradas digitais;
- 1 unidade de EL2809: Saídas digitais.

Antes de se efetuarem quaisquer configurações, é necessário:

- Ligar a rede fisicamente;
- Configurar a rede no *CX-Integrate*:

  - No lado do *KUKA*, retirar os ficheiros *EDS* (*Electronic Data Sheet*) e selecionar o controlador;

  - Efetuar o download dos ficheiros *EDS* correspondentes às unidades a configurar na rede (unidades de *BECKHOFF*).
- Configurar a rede no *WorkVisual*:

  - Efetuar o download dos ficheiros EDS correspondentes às unidades a configurar na rede;

  - Unidades de *BECKHOFF*:

    - BECKHOFF EtherCAT XML – XML Device Descrition;

    - EL6752-0010 – EtherCAT DeviceNet Slave.

3.3.1 Ligações físicas

A rede *DeviceNet* é constituída por resistências de terminação, isto é, no início e fim da rede (no módulo de comunicação DRM21 e no KR C4 R2) deve existir uma resistência de 121 Ω. Na Figura 3.15 encontra-se a ligação efetuada (resistência de 121 Ω) no sistema.

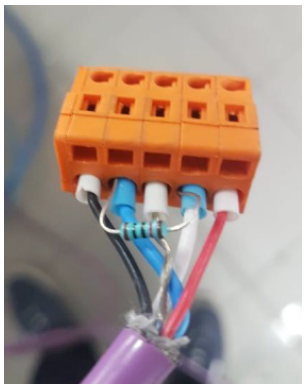


Figura 3.15: Conector *DeviceNet*

Na Figura 3.16, verifica-se que a ligação efetuada anteriormente encontra-se correta, sendo que o fio vermelho corresponde a +21V e o fio preto a 0V.

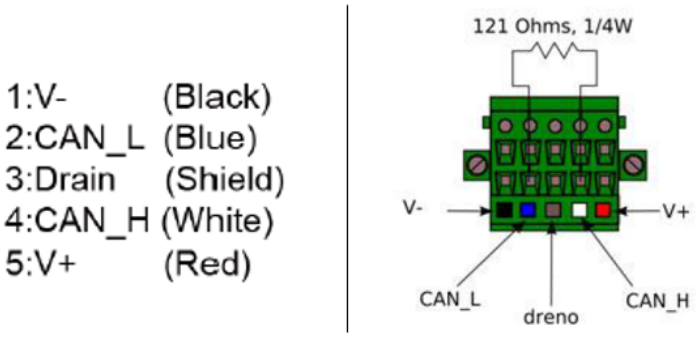


Figura 3.16: Ligações Conector DeviceNet

3.3.1.1 Unidade CJ1W-DRM21- Módulo *DeviceNet*

Este módulo é constituído por *DIP switches* com números de 1 a 4 e por três *switches* rotativos, apresentados na Figura 3.17.

Os *DIP switches* foram configurados da seguinte forma:

- 1 - OFF
- 2 - ON
- 3 - OFF
- 4 - OFF

Os *switches* rotativos foram configurados da seguinte forma:

- Unit No. - 1
- Node DR - 0 9

Assim, este módulo foi configurado para o endereço 9 com *baud rate* de 500.

Figura 3.17: Configuração unidade CJ1W-DRM21

3.3.1.2 Unidade BK5250 - Mesa de indexação

Para configurar esta unidade modificaram-se os dois *switches* rotativos para:

- Unit No. - 1 1

Assim, observando a Figura 3.18, esta unidade foi configurada para o endereço 11.

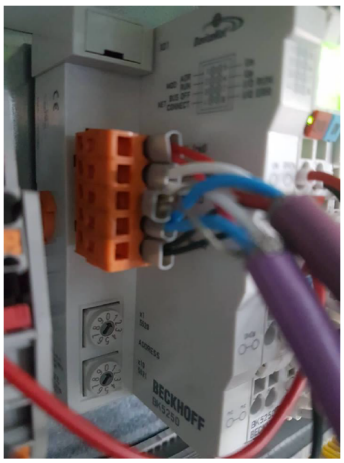


Figura 3.18: Configuração unidade BK5250 - Mesa indexação

3.3.1.3 Unidade KR - R1 e R2

O endereço destas unidades foi configurado através da ferramenta de trabalho *WorkVisual* sendo utilizados os endereços 13 e 15 para o robô R1 e R2, respetivamente.

3.3.2 Configurar rede

Como referido anteriormente, após a ligação física entre os componentes, foi necessário configurar o *PLC* e os controladores *KUKA*.

3.3.2.1 Configurar a rede no *CX-Integrate*

Para configurar a rede no *CX-Integrate* foi necessário efetuar o download dos ficheiros *EDS* correspondentes às unidades a configurar na rede. De seguida, foi necessário criar o *node* definido anteriormente, ou seja, o 9. Após isto, foram adicionados os *EDS* dos *KUKAS* inserindo os respetivos componentes. O *node* está de acordo com o definido anteriormente, 13 e 15. Na Figura 3.19 é mostrada a configuração final da rede.

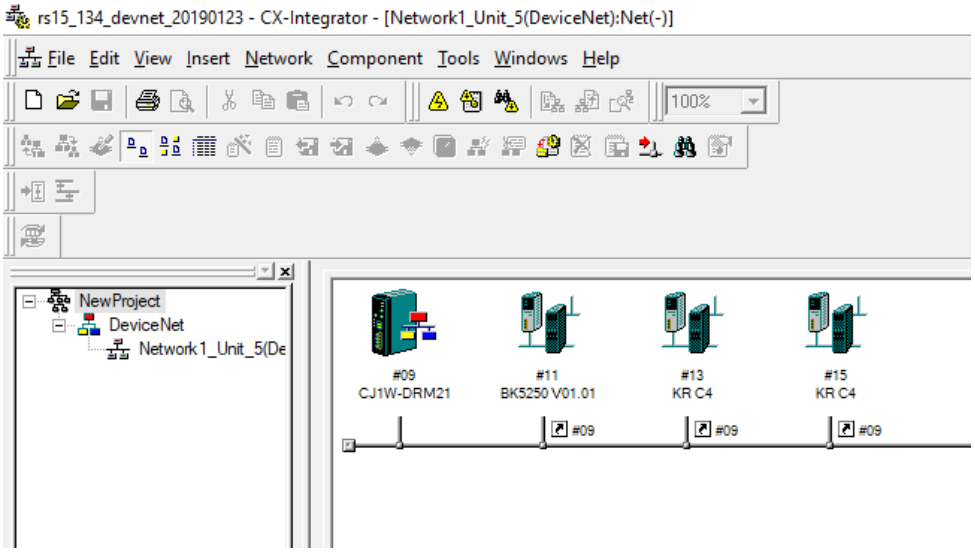


Figura 3.19: Configuração *CX-Integrate*

Um aspeto importante na configuração da rede de cada um dos controladores é o número de bytes que se deseja ter na saída *(OUT)* e na entrada *(IN)*. Neste caso, foi configurado com 64 bytes de entrada e de saída (*standard* da empresa *Robotsol*).

3.3.2.2 Configurar a rede no *WorkVisual*

Para configurar a rede no *WorkVisual* foi necessário ligar o *PC* ao controlador *KUKA*. Logo de seguida, utilizando o cabo *Ethernet*, foi necessário mudar o *IP* do computador. No *WorkVisual* foi efetuado o download dos ficheiros *EDS* já referidos, foi escolhido o controlador desejado e adicionaram-se os componentes físicos do controlador. Na Figura 3.20 é possível verificar a configuração final da rede.

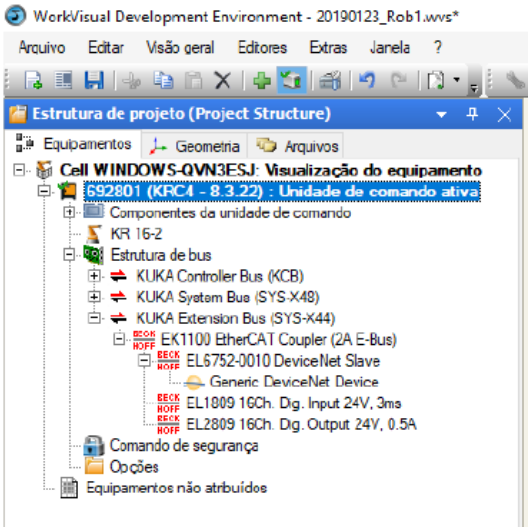


Figura 3.20: Configuração *WorkVisual*

Posto isto, configuraram-se os componentes adicionados anteriormente, adicionando-se o nome do robô, o endereço (deverá ser coincidente com o que se definiu do lado do *PLC*, 13 e 15) bem como o *baud rate* de 500 (deverá coincidir com o existente na placa DRM21).

Por fim, realizou-se o mapeamento de acordo com o que foi definido no *PLC*. Assim, o *DeviceNet* tem de entrada *(IN)* 512 bits (64 bytes) e saídas *(OUT)* 512 bits (64 bytes). Foi, ainda, possível associarem-se as diferentes entradas e saídas do *DeviceNet* do *Bus* de campo às entradas do KRC, sendo que as mesmas foram mapeadas de 0-511 bits.

As cartas de *BECKHOFF* foram configuradas com entradas de 16 bits (2 bytes) e saídas com 16 bits (2 bytes). Este processo foi realizado atribuindo as unidades de entrada e saída de *BECKHOFF* do *bus* de campo aos endereços do *KRC* das entradas e saídas, respetivamente. As entradas e saídas foram mapeadas de 601 - 616 bits.

3.4 Ligações físicas

Após realizar a comunicação *DeviceNet*, foi possível comunicar com a mesa de indexação e os controladores R1 e R2, mas ainda foi necessário realizar algumas ligações. Assim, ligou-se o *gripper* R1 e o *gripper* R2 aos seus controladores e o quadro da mesa de fixação ao *PLC*.

A nível do *gripper* do robô 1 efetuaram-se as ligações para a alimentação do sensor de pressão e sensores indutivos, para estes detetarem a posição das peças. Também se efetuaram as ligações necessárias para os sinais (eletroválvula monoestável para o vácuo, eletroválvula biestável para fechar e abrir a garra e sensores).

Para o robô 2 efetuaram-se as ligações para a alimentação do sensor de pressão e as ligações necessárias para os sinais (eletroválvula monoestável para o vácuo e sensor).

Na mesa de indexação existe um quadro com os sensores e eletroválvulas para atuarem os cilindros pneumáticos. Deste modo, efetuaram-se ligações diretamente do quadro da mesa para as entradas e saídas do *PLC* e verificou-se qual a eletróvalvula que correspondia a cada cilindro ou conjunto de cilindros pneumáticos.

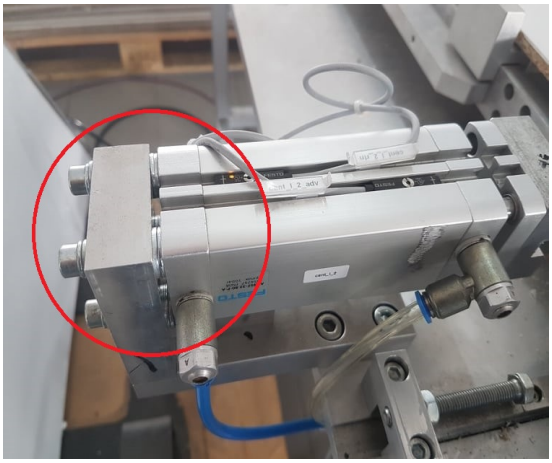
3.5 Melhorias na mesa de indexação

Para resolver o problema da célula, foi testada a mesa de indexação para, assim, confirmar que os quadros chegavam corretos e sem defeitos à mesa de fixação. O primeiro problema encontrado deve-se ao facto de existirem dois tipos diferentes de painéis. O primeiro tipo de painel é constituído por um perfil de papel de um lado e um perfil de metal do outro. As suas dimensões de 72,60 cm x 104,20 cm, variavam cerca de 0.02 cm. O segundo tipo de painel é constituído por perfis de metal nos dois lados. As suas dimensões não eram constantes, variando alguns centímetros entre si. Desta maneira, foi decidido pela *RobotSol* que apenas se usaria o primeiro tipo de painel.

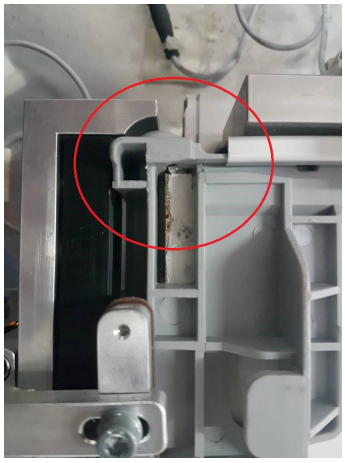
Neste subcapítulo foi testada a sequencia lógica, que se encontra no apêndice D, e verificou-se a existência de erros na mesa de indexação. É necessário relembrar que estes testes foram os primeiros a serem executados, servindo como forma de aprendizagem e possibilitando que algumas soluções encontradas sofressem alterações e melhorias. Ao seguir a sequência previamente referida, observou-se que se devem avançar todos os cilindros centradores. Verificou-se que existia uma folga entre o quadro e o centrador, Figura 3.21a.Para além disso, observaram-se outras falhas, como por exemplo, os cilindros dos topos com o painel inserido chegam ao fim de curso, não conseguindo centrar o quadro. Após uma série de testes, ficou claro que a mesa sofria falta de esquadria, o que provocava erros e pouco sucesso na construção dos quadros. Assim, foi necessário fazerem-se alterações mecânicas movendo os cilindros pneumáticos alguns milímetros. Como os cilindros pneumáticos estavam fixos à mesa, acrescentaram-se algumas anilhas, Figura 3.21b, que foram uma solução rápida e provisória aquando da fase de testes. Trocaram-se as anilhas por uma peça completa de forma a colocar os cilindros na posição correta. Verificou-se que, mesmo com essas alterações, não era possível um alinhamento perfeito. Porém, para evitar a construção de novas peças, alterou-se a regulação de ar nos centradores. Assim, os centradores dos topos ficaram mais rápidos do que os da base e os laterais da direita mais rápidos do que os da esquerda. Não foi uma boa solução, devido aos movimentos de torção (os cilindros pneumáticos não tinham a mesma força) e

existia a possibilidade de danificar os rolamentos existentes nas guias. A solução assentava em colocar os cilindros à distância correta e, para isso, seriam necessárias peças novas. De seguida, as peças metálicas foram colocadas corretamente (havia uma pequena folga que se conseguiu corrigir modificando a posição dos cilindros) faltando apenas colocar as peças de plástico. Aqui, foram encontrados mais dois problemas, Figura 3.21c. O primeiro, deveu-se ao facto do suporte que segurava a peça de plástico não a fixar totalmente, deixando uma folga. O segundo, residiu na falta de esquadria, levando a uma folga entre o painel e a peça de plástico. Este resultado, deveu-se ao facto de um lado entrar primeiro do que o outro, levando a que a peça não entrasse corretamente, pois não estava alinhada. Na Figura 3.21d, foi possível confirmar o mesmo problema para os suportes de base. As peças plásticas estavam desalinhadas, provocando desgaste e muitas vezes falhas.

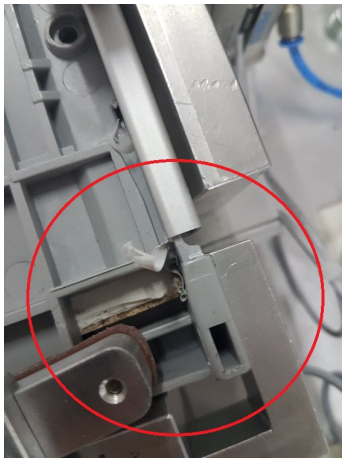
(a) Folga entre quadro e centrador



(b) Solução provisória



(c) Resultado do encaixe da peça de plástico topo



(d) Resultado do encaixe da peça de plástico base

Figura 3.21: Erros e melhorias na mesa de indexação

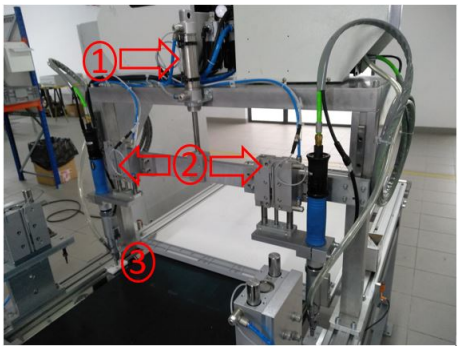
3.6 Melhorias na mesa de fixação

Neste subcapítulo, antes de aprofundar o tema principal, para perceber a sequência lógica aplicada que se encontra no apêndice E, é necessário explicar quais os nomes dados aos componentes utilizados. Assim, ao observar a Figura 3.22 , verificam-se as três zonas em que foi dividida a mesa de fixação.

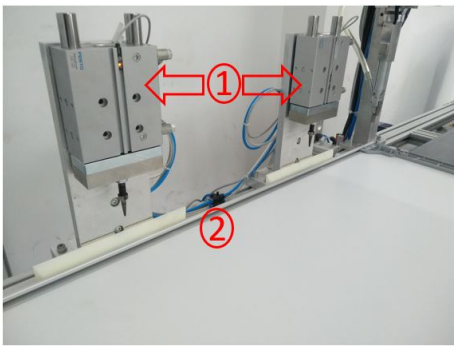
Na primeira zona representada na Figura 3.22a, os números um, dois e três, correspondem, respetivamente, ao 'cilindro topo aparafusadora', aos 'cilindros das aparafusadoras' e aos 'cilindros base aparafusadora'. É necessário ter em atenção que a eletroválvula que controla os cilindros da aparafusadora é a mesma. Isto é, quando se envia um sinal para avançar esses cilindros, os dois são avançados. O mesmo acontece com os 'cilindros base aparafusadora'.

Na segunda zona visível na Figura 3.22b, os números um e dois, correspondem, respetivamente, ao 'cilindro *push* base' e ao 'cilindro *push* topo'. Aqui, uma eletroválvula controla os quatro 'cilindros topo' (os visíveis na figura e os outros dois que se encontram no lado oposto). Os 'cilindros *push* base' têm o mesmo funcionamento citado anteriormente.

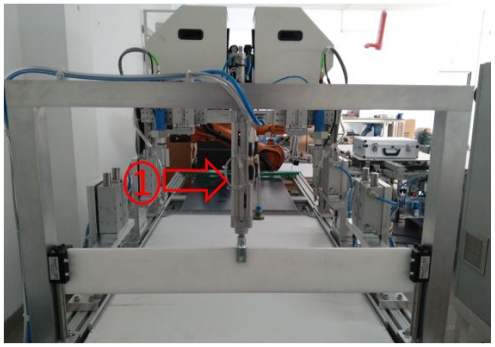
Por fim, a terceira zona corresponde à Figura 3.22c, sendo que o único número retrata o 'cilindro *stopper*'.



(a) Zona de aparafusamento



(b) Zona de amolgar



(c) Vista frontal do *stopper*

Figura 3.22: Detalhes dos Componentes da Mesa de Fixação

Após a explicação sobre a sequência lógica da mesa de fixação, encontrou-se o primeiro problema. O mesmo retratou a necessidade de haver material (painéis e peças) novo para se realizarem testes em condições iguais. No apêndice F, encontram-se todas as imagens retiradas para a perceção deste capítulo. Assim, encontraram-se dois problemas no material existente. O primeiro problema encontrado remeteu-se às peças metálicas e ao quadro, devido ao facto de terem sido perfurados em testes anteriores. Estas aberturas redirecionavam o movimento do parafuso, fazendo com que este se movimentasse para dentro delas, ficando torto e encravando no momento de aparafusar. O segundo problema encontrado deveu-se ao facto de alguns parafusos terem sido usados repetidamente, aquecerem e provocarem a deterioração dos mesmos. Os parafusos, na sua parte superior, podem ficar desgastados, provocando dificuldades a aparafusar, devido à ponta da aparafusadora não se encaixar e rodar por cima do parafuso. Também se encontrou desgaste na parte inferior provocando a perda da função de furar o metal. Assim, não foi possível ajustar o binário das aparafusadoras, devido à força exercida não ser constante (os furos existentes provocam menor resistência ao furar). A solução provisória encontrada foi cortar as laterais das peças de plástico para se realizarem testes ao longo do metal (que não tem furos). Após isto, colocou-se o binário no mínimo, sendo que, o mesmo, pode ser ajustado de 3 a 10. Imagens no apêndice F.

Mantendo as condições iguais nos materiais utilizados, foram observadas causas possíveis para o funcionamento irregular das aparafusadoras e da zona de amolgar. As causas possíveis do funcionamento irregular das aparafusadoras podem ser: desalinhamento do quadro; inclinação das aparafusadoras; binário utilizado; o quadro dobra com a força exercida pela aparafusadora. Na zona de amolgar, foi verificado, ainda, que os cilindros não conseguem dobrar o metal.

Para maior entendimento dos problemas e possíveis soluções, estes foram divididos em quatro etapas: alinhamento do quadro; funcionamento das aparafusadoras; testes de aparafusamento; amolgamento.

3.6.1 Alinhamento do quadro

Ao analisar a Figura 3.23, observa-se que C1 e C2 correspondem às medidas retiradas da mesa. Devido à simulação não estar atualizada e não corresponder à realidade, a seta indica a falta dos cilindros de base da aparafusadora.

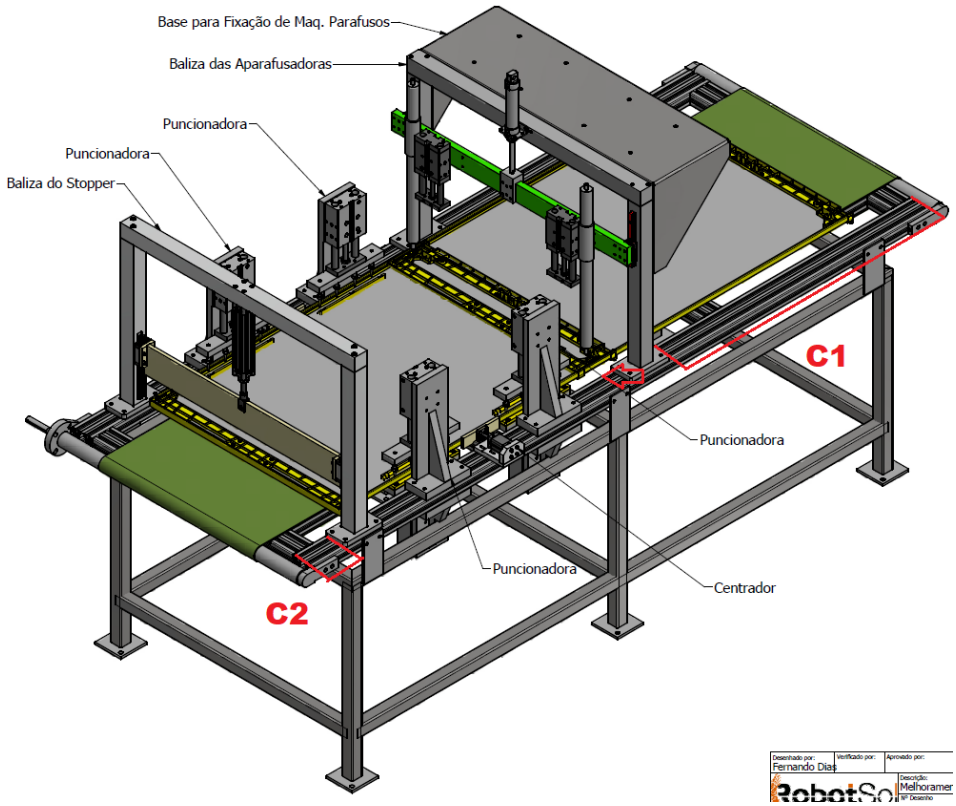


Figura 3.23: Características da mesa de fixação

- Mediram-se as seguintes distâncias:

  C1 lado direito: 99.5 cm

  C1 lado esquerdo: 99.2 cm

  C2 lado direito: 12.6 cm

  C2 lado esquerdo: 13.1 cm

Pela Figura 3.24a é visível um desfasamento entre a ponta da aparafusadora e o local onde se aparafusa. Desta forma, para se resolver este problema, utilizaram-se dois métodos.

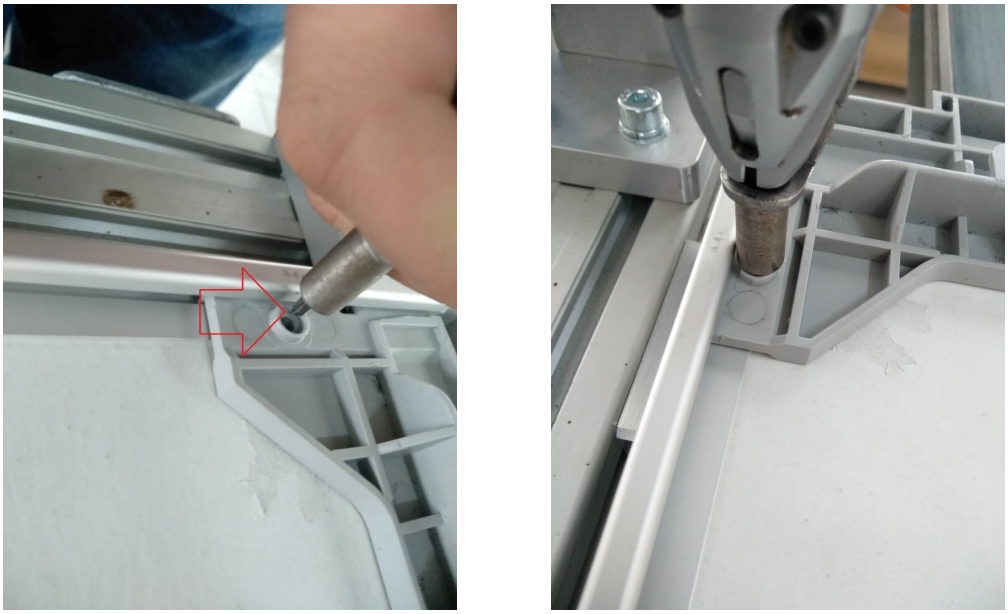
No primeiro método colocou-se o quadro na posição de aparafusar (como o robô) e ajustou-se a distância das aparafusadoras. De seguida, avançou-se o quadro até colidir com a barreira, ajustou-se 'C2' e verificou-se que as pontas das aparafusadoras estavam na posição correta. Por fim, o quadro voltou à posição inicial para confirmar se estava bem ajustado. Com este método não foi possível garantir ajustes, pois ao colocar o quadro na posição inicial a orientação do mesmo mudava. O quadro, quando colidia com a barreira, deslocava-se e perdia a orientação inicial.

No segundo método colocou-se o quadro na posição de aparafusar, mas mantendo a orientação do mesmo para todos os testes. Assim, foi criada uma base provisória que mantinha a orientação inicial. A partir daqui, o procedimento foi igual ao método anterior.

Por fim, a largura das aparafusadoras foi ajustada, mantendo-se as distâncias de 'C1' e mudando-se as medidas de 'C2' para:

- C2 lado direito: 13 cm
- C2 lado esquerdo: 12.8 cm

A figura 3.24b documenta que o alinhamento foi concluído com sucesso.



(a) Resultado inicial: Alinhamento incorreto **(b)** Resultado final: Alinhamento correto

Figura 3.24: Resultado do alinhamento

3.6.2 Testes de aparafusamento

Após o alinhamento correto do quadro, as aparafusadoras estavam à distância correta uma da outra, o binário colocado no mínimo possível e procedeu-se à verificação do ângulo das aparafusadoras em relação à mesa (90º). As mesmas estavam aproximadamente com a inclinação desejada, porém essa alteração não melhorou o seu funcionamento. Devido à base da aparafusadora não se encontrar em conformidade com a primeira solução, foi necessário testar se a mesma melhora o funcionamento.

Em primeiro lugar, testou-se com a base da aparafusadora recuada, Figura 3.25. Verificou-se que o quadro dobra consideravelmente, provocando a mudança da posição de aparafusamento. Esta fica inclinada tornando-se mais difícil de aparafusar.

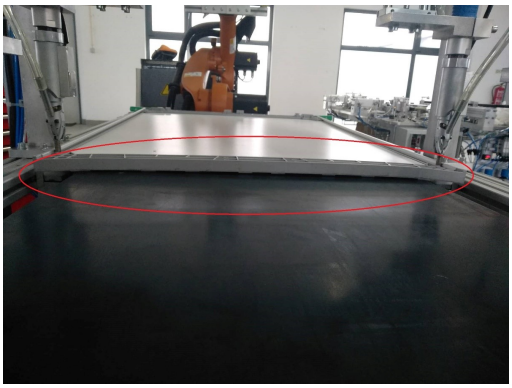
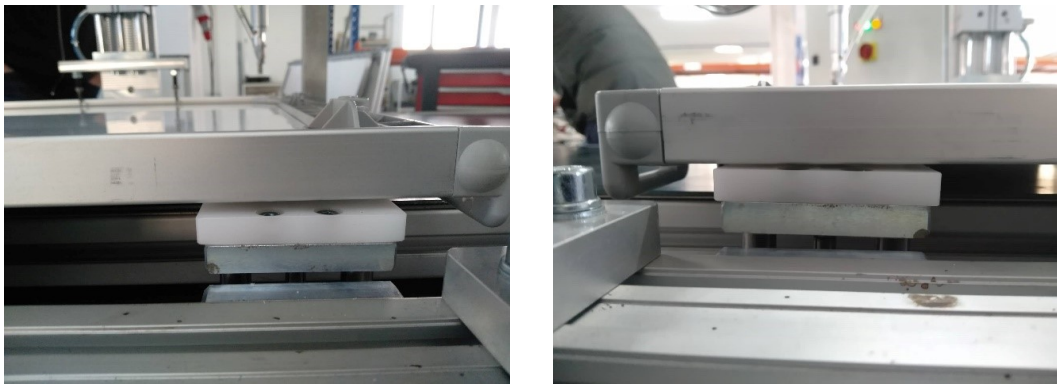


Figura 3.25: Base da aparafusadora recuado

Em segundo lugar, testou-se com a base da aparafusadora avançada, Figura 3.26. Assim, é possível verificar outros erros, nomeadamente, a base não toca no quadro, as bases das aparafusadoras estão desniveladas e as mesmas não suportam a força da aparafusadora, isto é, os cilindros descem à medida que o quadro é aparafusado.



(a) Base da aparafusadora avançado lado direito **(b)** Base da aparafusadora avançado lado esquerdo

Figura 3.26: Resultado base da aparafusadora

3.6.3 Funcionamento das aparafusadoras

De forma a descobrir a inconsistência das aparafusadoras e se compreender o seu mecanismo de funcionamento, estas foram desmontadas e analisadas. O local de aparafusamento fica próximo da peça de metal lateral, sendo que o espaço existente entre o metal e a ponta da aparafusadora é mínimo. Para aumentar o espaçamento desgastou-se a ponta da aparafusadora, Figura 3.24b. Após isto, descobriram-se três problemas. O primeiro retrata a ponteira da aparafusadora, que é a peça que se fixa ao parafuso. Esta estava desgastada, oscilava e tinha muita folga. O segundo problema, pode observar-se na Figura 3.27b. O círculo vermelho sinaliza uma peça lateral que serve para colocar os parafusos na posição correta,

quando estes descem a aparafusadora. Este problema consiste no facto da peça ter sido construída para um tipo de parafusos diferentes, o que provoca, por vezes, a mudança de direção do parafuso e o mesmo fica inclinado, aparafusando mal ou, até encravando. O terceiro problema encontrado, visa o funcionamento da máquina. Esta, para aparafusar, precisa ser empurrada contra o quadro. No entanto, existe um momento antes de aparafusar em que a máquina está parada na posição de aparafusamento. Nesse momento, abre-se uma peça na ponta da máquina e um parafuso desce para ficar na posição correta. É nesse momento que surge o problema. A altura a que máquina se encontrava não era a correta, isto é, ao avançar para a posição de aparafusamento (avançar com o cilindro pneumático topo) o parafuso não estava no local correto. Para ter um funcionamento correto, quando se avança para essa posição de aparafusamento, a mesma tem de ficar com o parafuso na posição correta e junto ao quadro. Na realidade, isto não se verifica, o que leva a que a máquina o tente aparafusar com o parafuso torto. Assim, a junção destes problemas, leva a que aparafusadora exerça uma força que a base não suporta. Os problemas retratados encontram-se nas duas aparafusadoras.

(a) Ponta da aparafusadora vista verticalmente **(b)** Ponta da aparafusadora vista horizontalmente

Figura 3.27: Ponta da aparafasudara alterada

3.6.4 Amolgar

Depois da zona de aparafusamento ter sido testada e melhorada, faltava testar a zona de amolgar os quadros. Pode-se ver na Figura 3.28a que o quadro não sofre qualquer deformação, apenas fica seguro pelas ponteiras.

Devido à falta de ineficácia da zona de aparafusamento, realizaram-se vários testes para melhorar esta parte da célula. As imagens dos testes realizados encontram-se no apêndice F.

No primeiro teste realizado, afastaram-se as pontas do cilindro pneumático em relação à largura do metal, tornando-se, assim, necessária uma menor força para dobrar o metal. Este teste não obteve

melhores resultados do que anteriormente.

No segundo teste realizado, trocaram-se e afiaram-se as ponteiras utilizadas anteriormente. O objetivo deste teste foi criar uma área de contacto menor, para assim curvar o metal. Os resultados obtidos não trouxeram melhorias.

No terceiro teste, usou-se uma ponteira que foi um pouco afiada, com o intuito de centrar toda a força do cilindro num único local. Verificou-se uma melhoria.

No quarto teste, afiou-se a segunda ponteira e colocaram-se todas as ponteiras exatamente à mesma altura e à mesma distância da lateral. Comparou-se com o primeiro teste e verificou-se que houve melhorias, mas estas ficaram longe dos resultados pretendidos.

Após esta série de testes retiraram-se algumas conclusões, nomeadamente:

- Afiar em demasia as ponteiras provoca a furação do metal e não amolga o mesmo.
- É necessária uma força pneumática maior.
- Há melhores resultados nos testes em que um cilindro pneumático exerce força com apenas uma ponteira.
- Há melhores resultados em testes em que as ponteiras exercem força no limite do metal.

Para não gastar orçamento em novos cilindros ou num repetidor de pressão, pensou-se em usar uma válvula de escape rápido. A ideia foi obter uma menor resistência da saída do ar e uma maior velocidade de descida, provocando deformação na chapa. Como não havia nenhuma válvula de escape rápido na empresa, retirou-se a válvula de escape do cilindro provocando uma menor resistência de saída do ar, simulando uma válvula de escape rápido. Desta maneira, foram executados uma série de testes.

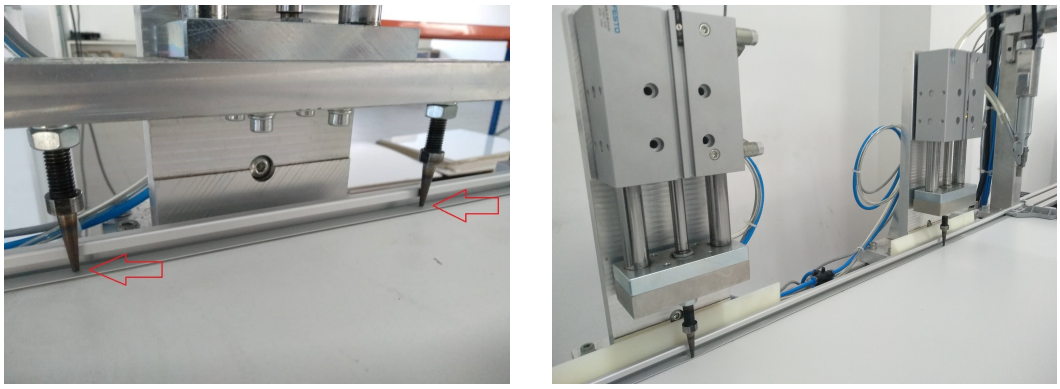
No primeiro teste, retirou-se a válvula de escape do cilindro e executou-se o amolgamento com os dois tipos de ponteiras existentes. Verificaram-se melhorias e as ponteiras iniciais obtiveram um melhor resultado.

No segundo teste, utilizou-se o conhecimento adquirido nos testes anteriores, isto é, usou-se apenas uma ponteira. Colocou-se a ponteira na zona central, retirou-se a válvula de escape do cilindro e criou-se uma ponteira em cunha. Testaram-se os três tipos de ponteiras individualmente, verificando-se os resultados com uma e duas batidas.

Após esta série de testes, retiraram-se as seguintes conclusões:

- Há melhores resultados com ponteiras normais.
- O resultado com duas batidas apresenta melhorias, mas perde tempo de execução.
- Uma ponteira no centro do cilindro, sem válvula de escape, provoca maior força e velocidade contra o quadro obtendo, assim, melhores resultados.

Na Figura 3.28b observa-se o resultado final dos testes realizados.



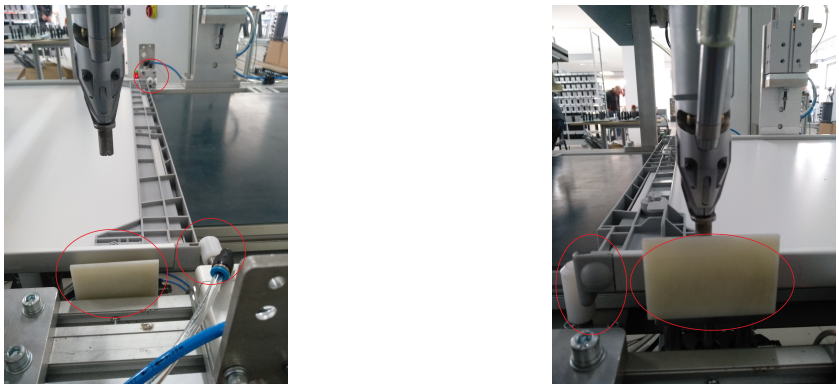
(a) Resultado inicial: Amolgar incorreto

(b) Resultado final: Amolgar correto

Figura 3.28: Resultado do alinhamento

3.6.5 Melhorias

Executados todos os testes e demonstradas algumas melhorias, falta apenas retratar uma modificação na mesa. A orientação inicial do quadro foi um problema demonstrado anteriormente. Como tal, a solução provisória foi modificada. Como uma ligeira inclinação do quadro provocava erros no aparafusamento, pensou-se numa solução que funcionasse para os casos mais extremos. Assim, retirou-se o *stopper* final e substitui-se por quatro *stoppers*. Como se pode observar na Figura 3.29, o robô não necessitava de colocar o quadro no ponto exato de aparafusamento, ganhando assim uma distância maior para posicionamento do mesmo. Posteriormente, o tapete empurrou o quadro contra os dois *stoppers* e subiu a base da aparafusadora. Como se pode reparar na Figura 3.29b, a base foi alterada por uma peça nova com um formato em 'L'. Esta nova base teve como objetivo delimitar a posição do quadro ficando o mesmo na posição correta. Por fim, os dois últimos *stoppers* foram utilizados como anteriormente, mantendo a posição de aparafusamento para a segunda posição de aparafusamento.



(a) Solução para a primeira posição de aparafusamento **(b)** Solução para a segunda posição de aparafusamento

Figura 3.29: Melhorias finais

3.6.6 Nova sequência lógica

Na sequência do que foi descrito explica-se o funcionamento final da mesa de indexação. Devido à mudança de cilindros pneumáticos na mesa, a sequência lógica foi alterada. Assim, o funcionamento começa com o robô a colocar o quadro na mesa. O sensor 1 é ativado e o tapete avança. O sensor 2 é ativado e começa a contar um *timmer*. Enquanto isso, os *stoppers* 1 encontram-se avançados fazendo com que o quadro colida com os mesmos. A base do cilindro sobe e o *timmer* chega ao fim e para o tapete na posição de aparafusamento. O passo seguinte é o aparafusamento, que é igual ao explicado na primeira sequência lógica. De seguida, a base desce e o *stopper* 1 recua, podendo o tapete avançar o quadro. O quadro avança até o sensor 1 deixar de o detetar. Nesse momento, o tapete para, os *stoppers* 2 avançam e o tapete recua levando o quadro a colidir com os *stoppers*. Quando o sensor 1 deteta novamente o quadro, outro *timmer* é ativado. A base da aparafusadora sobe e o *timmer* acaba, provocando a paragem do tapete. Com isto, o quadro está na segunda posição para aparafusamento. O procedimento é igual ao explicado na primeira sequência. Logo após o aparafusamento, o quadro é amolgado. Posteriormente, a base da aparafusadora desce, os *stoppers* 2 recuam e o quadro avança para o tapete seguinte.

A nova sequência lógica encontra-se no apêndice G. É de salientar que cada modificação da sequência envolve a necessidade de modificar o código do *PLC*.

3.7 Programação Robô 2

Nesta secção, o código final desenvolvido é explicado através de fluxogramas. Estes fluxogramas retratam a lógica de programação, sem referir especificamente as variáveis utilizadas. Como referido

anteriormente, a linguagem de programação é a *KRL - KUKA Robot Language*. O documento de treino *KUKA*, [18], foi lido e utilizado para a aprendizagem da linguagem e lógica aplicada. Inicialmente, apresenta-se e descreve-se o programa principal. De seguida, aborda-se o subprograma de *pick* e o subprograma de *place* do quadro montado.

O fluxograma do programa principal encontra-se representado na Figura 3.30. O programa principal começa por inicializar as condições iniciais, isto é, a sua ferramenta não liga a sucção e usa variáveis específicas do robô para informar o *PLC* no estado em que se encontra, tanto a nível de erros, como a nível programas utilizados, áreas, processos e colisões. Seguidamente, é executado o código responsável pela movimentação do robô para uma posição segura ("*Home*"). Caso o robô não se encontre nessa posição é enviado um aviso para se mover. Nesse momento, o robô encontra-se pronto para executar qualquer programa, portanto, só é necessário existir um programa, mas o robô pode ter vários para tarefas distintas. Assim, o robô envia um pedido de código do programa para o *PLC* e permanece à espera da resposta do mesmo, até receber um código do programa. Caso o seu valor seja zero, o código permanece preso em *loop* até receber outro valor. Caso seja o número do programa correspondente, o mesmo sai do *loop*. De seguida, avança-se para o acesso à célula. Primeiramente, recebe-se o pedido de acesso à célula, este é concedido e avança-se para o subprograma *Table_pick*. O código referido no momento seguinte (subprograma intermédio) visa a deslocação do quadro até à mesa de fixação. Neste caso, são pontos intermédios entre a mesa de indexação e fixação. Os mesmos podem ser observados no apêndice H. Os pontos 1, 2 e 4 têm um tipo de movimento *LIN* (linear), isto é, têm movimentos de trajetória em linha reta. O ponto 3 tem um tipo de movimento PTP (*Point to point*- Ponto a ponto), isto é, o robô conduz ao longo da trajetória mais rápida até ao ponto de destino. Normalmente, a trajetória mais rápida não é a trajetória mais curta e, portanto, não é uma reta. Devido aos eixos do robô se movimentarem rotativamente, as trajetórias em curva podem ser executadas com maior rapidez do que trajetórias retas. O problema de usar este tipo de movimento é que o percurso exato não é previsível. Neste movimento, utilizando uma propriedade de comando (*CONT*), é possível fazer uma aproximação ao ponto escolhido, trazendo vantagens quer a nível de desgaste, quer a nível de tempos menores de cadência. A desvantagem é que o robô não passa diretamente em cima do ponto escolhido.

A trajetória que o robô executa começa no ponto 1, 2, 4 e regressa pelo ponto 3. Com a explicação anterior, percebe-se que o robô, quando tem um quadro, necessita de percorrer os pontos marcados, de forma a não colidir. No seu regresso, o robô tem maior liberdade, sendo possível fazer uma aproximação ao ponto e, assim, poupar tempo de ciclo. Posteriormente, o subprograma *Conveyor_Place* é executado e retorna só com a ferramenta, utilizando os pontos intermédios referidos anteriormente (subprograma

intermédio_volta). Termina-se com o "Programa concluído". Nesta fase, o robô envia para o *PLC* a informação de que o programa foi terminado. Caso receba a resposta de que o código do programa terminou, move-se para a posição *Home*, voltando ao início do programa. Caso o código do programa não termine, volta para o "Pedido de programa", continuando a produção.

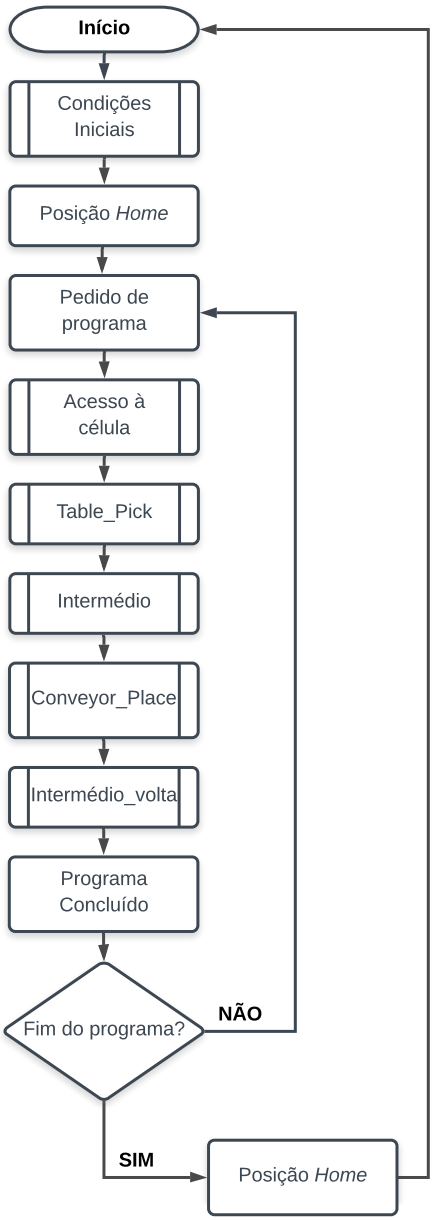


Figura 3.30: Programa principal do robô 2

Na sequência apresentada anteriormente, explica-se o fluxograma do subprograma *Table_pick* que se encontra apresentado Figura 3.31. Como o nome indica, o programa é iniciado na mesa de indexação quando o quadro está montado. O subprograma começa por realizar o pedido de área, isto é, o robô comunica com o *PLC* para obter informação acerca da área. Caso a área esteja ocupada pelo outro robô,

o mesmo espera pela ordem do *PLC*. Quando a área estiver livre, o robô 2 move-se para a posição de pegar (*pick*) no quadro. Como referido anteriormente, foi escolhido um ponto para a mesa de indexação, que se encontra junto ao quadro. Para o robô não fazer o movimento diretamente sobre esse ponto, foi aumentada a distância positivamente no eixo z em relação ao ponto marcado. A necessidade de utilizar uma única coordenada e, através dela, delimitar o percurso do robô, deve-se a dois fatores. O primeiro fator remete ao facto do caminho delimitado ser conhecido, sendo, previamente, possível programar o tipo de movimento escolhido, quantos pontos são necessários em relação ao ponto principal, entre outras variáveis. Isto provoca menos erros na marcação de pontos. O segundo fator deve-se ao tempo de marcação de pontos, ou seja, menor quantidade de pontos principais é sinónimo de menor quantidade de marcações de pontos, logo, menos tempo gasto.

Quando o robô chega à posição de '*picking*', o mesmo dá um aviso de que está a pegar no quadro. De seguida, liga o vácuo. A função de ligar o vácuo ativa a variável responsável por colocar o vácuo *on*. Posteriormente, utiliza um *timmer* e espera que a variável que corresponde ao sensor de pressão seja ativada. Caso a variável, no fim do *timmer*, não esteja ativada, ocorre um erro e o robô tenta pegar no quadro outra vez. Caso esteja tudo correto, é feito um pedido de processo. Este pedido acontece quando duas máquinas trabalham em conjunto. Neste caso, antes de o robô poder levar o quadro, é necessário desligar o vácuo da mesa de indexação. Assim, após desligar o vácuo e recuar os posicionadores de vácuo, o processo é concluído. O robô avança para o ponto seguinte e é libertada a área. Por último, o subprograma chega ao fim e volta ao programa principal.

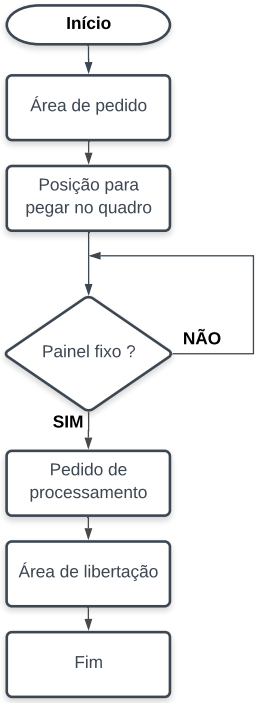


Figura 3.31: Subprograma *pick* do quadro completo

De seguida, explica-se o fluxograma do subprograma *conveyor_place* que se encontra representado na Figura 3.32. Este subprograma começa depois do robô chegar ao ponto 2, ver apêndice H. Assim, é necessário realizar o pedido de área. O funcionamento do pedido de área é igual ao referido anteriormente. Posteriormente, o robô posiciona-se para colocar o quadro na mesa de fixação. Quando este se movimenta até o sítio correto, é mostrada uma mensagem a indicar que o robô se encontra na posição certa. Seguidamente, verifica-se se o painel foi perdido, ou não. Caso a variável responsável do sensor de pressão não esteja ativa, esta indicará que o quadro foi perdido. Se assim for, é emitido um erro associado a este acontecimento, é desligado o vácuo, o operador tem que levar o robô para a posição *Home* e há uma libertação na área em que o mesmo se encontrava. O subprograma retorna ao início do programa principal. Caso o painel não esteja perdido, será desligado o vácuo. A função que desliga o vácuo é igual à explicada neste capítulo, sendo a única diferença de desligar em vez de ligar o vácuo. Assim, caso o sensor de pressão indique que o quadro continua preso, existe uma nova tentativa de libertação do mesmo. Caso o painel seja libertado, o robô desloca-se para os pontos seguintes, comunica com o *PLC* e a área da mesa de fixação é libertada. Por último, o subprograma chega ao final e volta ao programa principal.

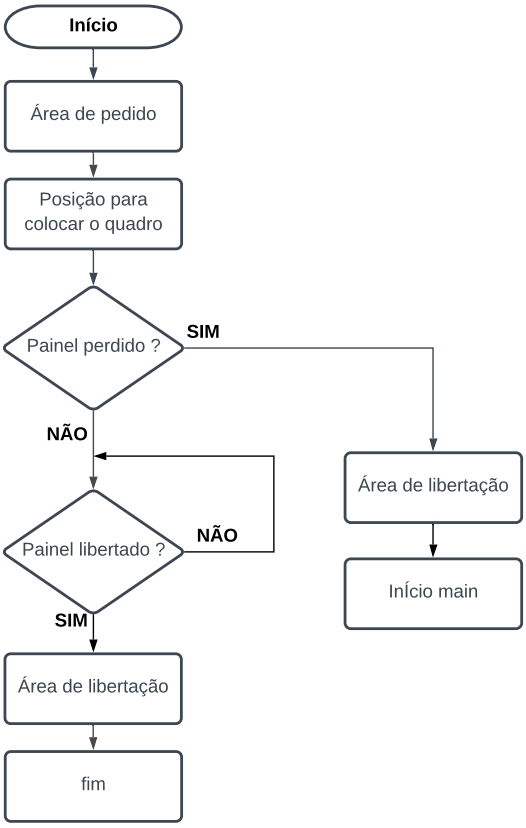


Figura 3.32: Subprograma *Place*

Terminada a explicação da sequência do robô, é necessário relembrar que o mesmo é o *slave* (escravo) e o *PLC* é o *master* (mestre), ou seja, o robô espera sempre pelas ordens que o *PLC* fornece. Os sinais digitais configurados e utilizados na programação do robô 2 encontram-se no apêndice H. A aba 'Notes', visível na Figura H.2, retrata o endereço utilizado no *PLC* correspondente ao sinal do robô 2. Como referido anteriormente, a nível da configuração da comunicação *DeviceNet* no robô 2, utilizam-se 64 bytes de entrada e saída. As saídas no robô 2 começam no endereço 3244 e as entradas no endereço 3349.

3.8 Programação *PLC*

Nesta secção demonstra-se o porquê de o *PLC* ser o controlador de todo o processo. Este é quem controla os robôs, as mesas, o *HMI*, entre outros. Deste modo, o controlador tem a função de gerir todos os sinais de entrada e ativar os sinais de saída. Assim, sem um bom controlo e uma boa gestão, a célula pode ter erros. Para se evitar qualquer tipo de erro, pensaram-se sempre em todos os cenários possíveis, assim, caso algum aconteça, este é identificado e resolvido o mais rapidamente possível. Para

além disso, este subcapítulo mostra a lógica utilizada, a estrutura do código desenvolvido e apresenta a evolução da programação efetuada, desde o primeiro código até à última versão.

Antes da explicação do código desenvolvido, recorda-se que na programação de um *PLC* não se devem utilizar múltiplas vezes a mesma saída, isto é, a saída que corresponde a um endereço lógico é criada uma vez e pode ser atuada múltiplas vezes. Para tal, é necessário utilizarem-se variáveis auxiliares de trabalho (W) com o formato *BOOL* (variável de um bit, os estados possíveis são 0-OFF e 1-ON). O seu funcionamento é simples, por exemplo, coloca-se no local onde se quer atuar um cilindro pneumático, a variável auxiliar como *saída* (*OUT*). No local onde existe a *saída* especifica desse cilindro pneumático, coloca-se a variável auxiliar como *entrada*. Assim, quando a variável auxiliar *saída* é ativada, atua a variável auxiliar *entrada* que, por sua vez, atua a saída, que é o cilindro pneumático.

Após uma breve explicação do funcionamento das varáveis, observa-se na Figura 3.33a a estrutura do primeiro código realizado. Este programa desenvolveu-se para testar a mesa de fixação, contendo apenas um subprograma constituído pela inicialização da mesa (*Init*), isto é, utilizam-se variáveis auxiliares para colocar os cilindros recuados ou avançados de acordo com o pretendido. De seguida, é feito o controlo da mesa (*Mesa2*). Esta é o local onde se programa a sequência lógica, isto é, avança-se o tapete rolante, recuam-se cilindros, configuram-se *timmers*, entre outras coisas. Por fim, é visível o local onde atuam as variáveis de saída (*Outputs*). Também é possível observar-se o bloco de funções (*Function Blocks*). Como o nome indica, é uma função constituída por vários blocos, que utiliza variáveis de entrada e variáveis de saída. Para poupar repetição de código, foi criada uma função de blocos correspondente a um cilindro pneumático. O seu funcionamento é explicado mais à frente.

Na Figura 3.33b encontra-se a estrutura do último código desenvolvido. O mesmo, é dividido em quatro subprogramas, *Inputs*, *Main*, *Robots* e *Outputs*.

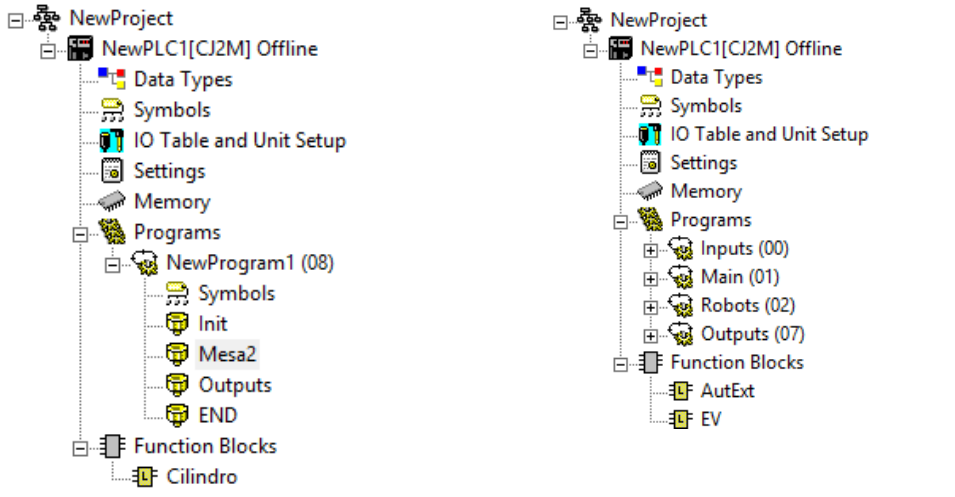
Passa-se, então, à explicação do primeiro subprograma,*Inputs*. Como indicado no nome, são todas as entradas de sinais provenientes de toda a célula. As mesmas são provenientes do robô 1, do robô 2, da mesa de indexação, da pilha de entrada de painéis, do transportador de entrada de acessórios, da mesa de fixação, do *HMI*, dos botões da célula (botões do quadro elétrico, botões das *psengates*, botões da barreira do armazém e estados dos relés de segurança) e da transportadora de saída. Em relação a este subprograma, o seu funcionamento retrata a conexão do sinal de entrada com da variável de trabalho (W) associada à mesma. Todas as entradas, quer ligadas diretamente nas cartas do *PLC*, quer através do sistema de comunicação *DeviceNet*, são representadas por uma variável de trabalho ao longo do programa. Tal, quer dizer que se modifica uma entrada física no *PLC*, isto é, retira-se um fio na carta de entrada e muda-se para outra (muda o endereço associado ao sinal). Caso não exista uma

variável de trabalho (W) associada, é necessário alterar a variável de entrada ao longo do código, todas as vezes que se utilize esse endereço. Com uma variável associada, basta alterar o endereço no código, uma vez.

O subprograma *Robots* é constituído por quatro subprogramas e cada robô têm dois subprogramas associados. O primeiro corresponde ao modo automático externo. Assim, utiliza-se um bloco de funções criado para este efeito tendo o robô a obrigatoriedade de ativar neste modo, caso os relés de segurança estejam ativos e seja clicado no botão verde. Caso o robô esteja neste modo, essa informação será enviada para o *HMI*, possibilitando ao operador tomar conhecimento da mesma. O segundo subprograma corresponde ao controlo do robô. Este controlo envolve a comunicação com o robô, visto no subcapítulo anterior, e a paragem do robô associado quando a célula é parada (clicar três segundos no botão vermelho).

Quanto ao subprograma *Outputs*, são todas as saídas de sinais provenientes do *PLC* para toda a célula. As mesmas, são enviadas para o robô 1, robô 2, mesa de indexação, pilha de entrada de painéis, transportador de entrada de acessórios, mesa de fixação, *HMI* e transportadora de saída. O seu funcionamento foi referido anteriormente.

Devido ao subprograma *main* ser mais complexo, o seu funcionamento é explicado posteriormente. Os dois blocos de funções criados são, também, explicados de seguida.



(a) Estrutura do primeiro código desenvolvido **(b)** Estrutura do último código desenvolvido

Figura 3.33: Evolução da estrutura do código do PLC

O código desenvolvido percorre uma série de etapas desde a primeira solução da mesa de fixação até à última. Foi feita a comunicação com os robôs e todo o controlo da célula. É necessário relembrar que o código realizado foi alterado sempre que algum componente foi modificado. Os dois tópicos da estrutura,

referidos anteriormente, são explicados no subcapítulo seguinte.

3.8.1 *Function Blocks*

Na programação do *PLC* criaram-se dois blocos de funções.

O primeiro bloco refere-se ao controlo de cada cilindro pneumático, isto é, cada *function block* criado possui um nome associado e controla um cilindro pneumático específico. O controlo envolve o avanço ou recuo do cilindro, bem como o estado em que se encontra o mesmo, ou seja, utilizam-se sensores para confirmar a posição do cilindro. Também controla erros de avanço ou recuo. O bloco é constituído por *inputs* e por *outputs*.

Os *inputs* são formados por variáveis do tipo *bool* (0 ou 1) e do tipo *uint* (inteiro). As primeiras são compostas pela ordem de avanço e recuo pelos sensores de avanço e recuo. As segundas controlam os tempos de avanço, recuo e erro. Os tempos de avanço e recuo controlam o tempo em que a variável de saída é ativada, isto é, ao avançar um cilindro pneumático pode ser controlado o tempo em que a variável de avanço ou recuo fica ativa. O tempo de erro é o tempo máximo que a variável de avanço ou recuo está na mesma posição sem avançar para a posição pretendida (o sensor não é ativado).

Os *outputs* são constituídos por variáveis do tipo *bool* (0 ou 1). As variáveis são as saídas do sistema após o controlo, sendo que as mesmas podem recuar ou avançar o cilindro pneumático. Informam o estado do erro, quer seja erro de avanço, quer seja erro de recuo e o estado dos tempos de avanço. Por fim, quando uma electroválvula atua vários cilindros pneumáticos, e alguns não possuem sensores de avanço e recuo, aplica-se no bloco de funções correspondente ao cilindro pneumático que tem a falta de um sensor, o sensor correspondente a outro cilindro pneumático. A vantagem da utilização deste método é a redução de custos extra, devido à compra de sensores. A desvantagem remete-se ao facto de ocorrer um problema no cilindro pneumático que não possui um sensor, não sendo detetado o erro e podendo provocar problemas na célula.

O segundo bloco refere-se ao controlo do modo automático externo dos robôs, isto é, cada robô possui um *function block* com um nome associado, que controla e gera os sinais da forma mencionada. O controlo envolve o início automático do sistema e operação normal, utilizando dois métodos diferentes. O primeiro método, é a confirmação do programa por meio de um sinal específico do robô (*PGNO_VALID*), o segundo retrata a confirmação do programa por meio de outro sinal específico do robô (*EXT_START*). Também é gerido o reinício do robô quando o mesmo é exigido que seja parado. Identificam-se vários modos de paragem, nomeadamente, a travagem dinâmica (*Operator safety*), a paragem de emergência (*EMERGENCY STOP*), a paragem após o *Motion enable* e a paragem depois de se utilizar o STOP.

Os sinais podem ser vistos no anexo B bem como o controlo necessário a ser realizado. O documento de treino *KUKA*, [36], foi lido e consultado para aprendizagem dos sinais específicos do robô utilizado, bem como os seus significados e funcionamentos.

3.8.2 Estrutura do programa principal - *Main*

Na estrutura do programa principal (*main*) realizaram-se vários subprogramas correspondentes aos componentes utilizados na célula. O diagrama encontra-se representado na Figura 3.34.

Observa-se a estrutura e verifica-se que o primeiro subprograma se intitula de "*Start Control*", controlo inicial. Este, como o nome indica, controla o arranque da célula. Para começar o arranque é necessário pressionar o botão verde, depois, realizam-se verificações de condições do sistema. Assim, é verificada a segurança da célula: se a pressão do ar está correta, se o armazém de painéis está preparado (recebe do subprograma armazém essa confirmação), se a mesa de indexação está pronta (recebe do subprograma mesa de indexação essa confirmação), se o transportador de entrada está pronto (recebe do subprograma transportador de entrada essa confirmação), se a mesa de fixação está pronta (recebe do subprograma mesa de fixação essa confirmação), se o transportador de saída está pronto (recebe do subprograma transportador de saída essa confirmação) e se os robôs estão na posição "*home*", no modo automático externo e prontos. Com isto, ainda se controla o fim da produção e, caso se force a paragem da célula, o programa volta ao início. Se alguma condição não for verificada, é dado o erro associado e indicado no *HMI*.

De seguida, encontra-se o subprograma "*Cell*", célula. Com este controlam-se erros e as luzes de sinalização da célula. Caso a célula esteja parada, é ativada a luz vermelha, mas se ocorrer o stop de emergência ou erro, a mesma tem uma luz intermitente vermelha e o *buzzer* é ativo. Caso a célula esteja em andamento, é ativada a luz verde e, se a célula está para parar, é ativada a luz verde, intermitentemente. Após uma paragem de emergência, é ativada uma luz laranja e é necessário o *reset* das seguranças. Este, poderá ser feito no armário do sistema pressionando o botão "rearme". Caso existam avisos, uma luz branca é ativada. Se algum operador necessitar de entrar na célula, é necessário clicar no botão azul de uma das portas de entrada do sistema, que tem um equipamento de trinco denominado *PSENsgate*. Após este pedido de entrada da célula, é ativada a luz azul, de modo intermitente. Caso o pedido seja aceite, a luz azul fica fixa e é possível entrar na célula. Por fim, ainda se controlam os tempos da célula, isto é, o tempo que a linha está em automático, de porta aberta, em alarme e sem material. Neste subprograma, também existe comunicação com o *HMI*, onde se enviam vários estados e erros.

Os três subprogramas "armazém de painéis", "mesa de indexação"e "transportador de entrada"não

foram desenvolvidos nesta dissertação.

Desta forma, explica-se o subprograma "mesa de fixação". Este começa com o controlo dos botões (verde ou vermelho) para início ou paragem do sistema. Caso seja pressionado o botão verde, o programa começa pelos estados iniciais e coloca todos os componentes da mesa na posição inicial correta. Após essa etapa, começa o controlo da mesa. Assim, existe toda a comunicação realizada com o robô 2, bem como a sequência da mesa. De modo a esclarecer o funcionamento do subprograma "transportador de saída", este realiza o controlo do tapete de saída com o sensor existente.

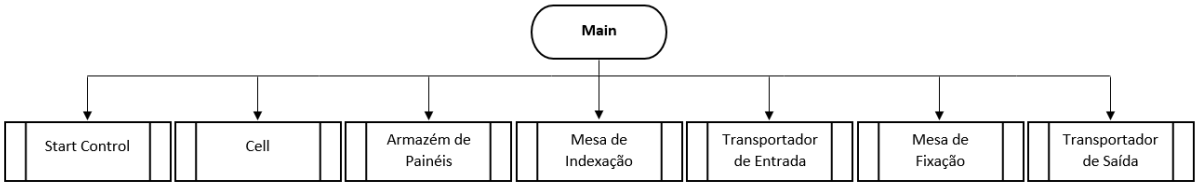


Figura 3.34: Estrutura do programa *Main*

3.9 Programação HMI

Neste subcapítulo, é feita a descrição da funcionalidade de algumas janelas mais importantes da consola *HMI*. Utilizou-se uma programação por blocos e foi programada através do software NB Designer da *OMRON*. São, também, explicados os principais ecrãs criados, identificando cada componente e a sua funcionalidade.

Na consola *HMI*, conseguem-se criar utilizadores, apresentar dados estatísticos e associar botões e lâmpadas a registos do *PLC*. Também se monitorizam sinais e parâmetros do processo da célula. Algumas ações da consola requerem um nível mínimo de utilizador. Esta consola não se encontra no seu estado final, necessitando de alguns ajustes e melhorias. Assim, na Figura 3.35 é possível analisar os dados de produção da célula, nomeadamente, os número de quadros produzidos, o tempo total e o tempo de ciclo. Também é possível realizar um *reset* aos contadores de produção e visualizar o estado do sistema.

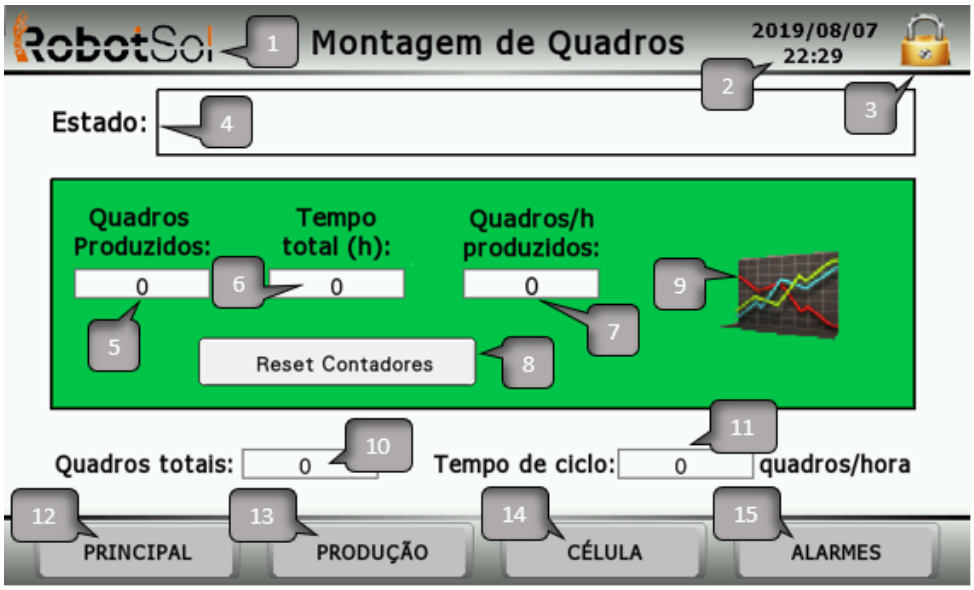


Figura 3.35: Ecrã principal da consola *HMI*

Legenda da Figura 3.35:

1. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de contactos.
2. Permite visualizar a hora e data atual. Ao pressionar, o utilizador é direcionado para a janela de edição de data e hora.
3. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização que permite mudar de utilizador.
4. Indica o estado da célula.
5. Indica o número de quadros produzidos.
6. Indica o tempo total (horas).
7. Indica o número de quadros produzidos por hora.
8. Reinicia as três últimas informações referidas anteriormente.
9. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de estatísticas (gráficos e tabelas).
10. Indica o número de quadros totais.
11. Indica o tempo de ciclo.
12. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização principal.
13. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de produção.
14. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização dos estados da célula.
15. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de alarmes.

Na Figura 3.36 é possível visualizarem-se os estados da célula, isto é, qual o estado em que esta se encontra (parado, erro, emergência, etc). Também é possível observarem-se as zonas correspondentes da célula, bem como os seus estados.

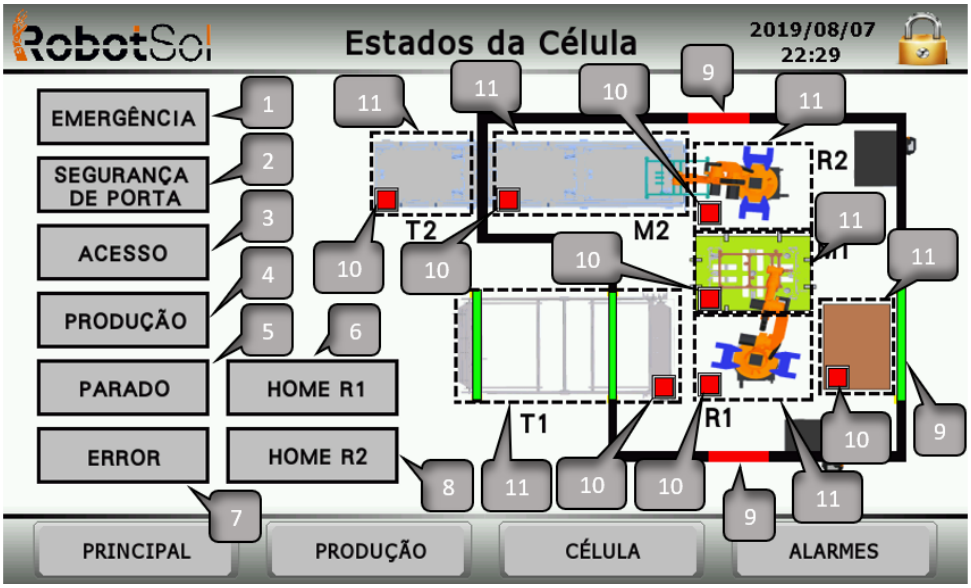


Figura 3.36: Ecrã correspondente aos estados da célula

Legenda da Figura 3.36:

1. Indica o estado de emergência da célula (ativo torna-se vermelho, inativo cinzento).
2. Indica o estado de segurança de porta (ativo torna-se vermelho, inativo cinzento).
3. Indica o estado de acesso da célula (ativo torna-se azul, inativo cinzento).
4. Indica o estado de produção da célula (ativo torna-se verde, inativo cinzento).
5. Indica o estado de paragem da célula (ativo torna-se vermelho, inativo cinzento).
6. Indica o estado de posição *Home* do robô 1 (ativo torna-se verde, inativo cinzento).
7. Indica o estado de erro da célula (ativo torna-se vermelho, inativo cinzento).
8. Indica o estado de posição *Home* do robô 2 (ativo torna-se verde, inativo cinzento).
9. Indica o estado das portas de entrada (ativo torna-se verde, inativo vermelho).
10. Indica o estado da área correspondente (ativo torna-se verde, inativo vermelho).
11. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização correspondente à zona selecionada.

Ao pressionar na área 'T2', o utilizador é direcionado para a janela do transportador de saída, Figura 3.37. Nesta, é possível avançar ou recuar o tapete, bem como observar o estado do sensor (sensor que deteta se existe quadro completo ou não). Caso o sensor esteja a verde, significa que o mesmo se

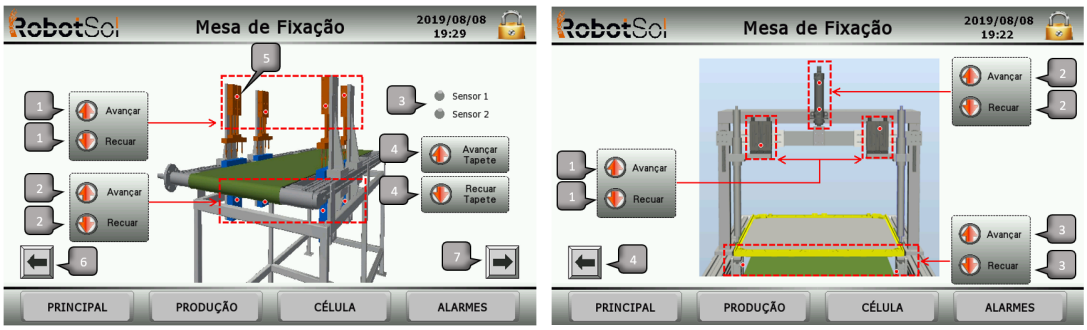
encontra ativo. Caso esteja cinzento, significa que se encontra inativo. Ainda se pode regressar para a janela de estados da célula, pressionando um botão. Por fim, verifica-se que a imagem está desatualizada e é necessário modificar a mesma.

Figura 3.37: Ecrã correspondente à zona do transportador de saída

Legenda da figura 3.37:

1. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de estados da célula.
2. Indica o estado do sensor (ativo torna-se verde, inativo cinzento).
3. Ao pressionar no ecrã, o utilizador avança ou recua o tapete desta área.

Ao pressionar na área 'M2', o utilizador é direcionado para a janela de amolgar da mesa de fixação, Figura 3.38a. Nesta, é possível avançar ou recuar o tapete, avançar ou recuar os cilindros pneumáticos, bem como observar o estado dos sensores (sensores que detetam o quadro e sensores que detetam a posição dos cilindros pneumáticos). Caso algum sensor dos cilindros pneumáticos esteja a verde, significa que o mesmo se encontra ativo. Caso esteja vermelho, significa que se encontra inativo. Ao pressionar um botão regressa-se para a janela de estados da célula ou avança-se para a zona de aparafusar. Caso avance para a zona de aparafusar, Figura 3.38b, o utilizador avança ou recua cilindros pneumáticos. Ao pressionar um botão, o utilizador regressa para a janela da zona de amolgar. Esta imagem encontra-se desatualizada e é necessário atualizarem-se os *stoppers*. Por fim, pode observar-se que cada cilindro tem o número de sensores correspondentes à realidade.



(a) Ecrã correspondente à zona de amolgar da mesa de fixação

(b) Ecrã correspondente à zona de aparafusar da mesa de fixação

Figura 3.38: Evolução da estrutura do código do PLC

Legenda da Figura 3.38a:

1. Ao pressionar no ecrã, o utilizador avança ou recua os cilindros pneumáticos topo.
2. Ao pressionar no ecrã, o utilizador avança ou recua os cilindros pneumáticos base.
3. Indica o estado dos sensores (ativo torna-se verde, inativo cinzento).
4. Ao pressionar no ecrã, o utilizador avança ou recua o tapete desta área.
5. Indica o estado dos sensores (ativo torna-se verde, inativo vermelho).
6. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de estados da célula.
7. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização da zona de aparafusar da mesa de fixação.

Legenda da Figura 3.38b:

1. Ao pressionar no ecrã, o utilizador avança ou recua os cilindros pneumáticos da aparafusadora.
2. Ao pressionar no ecrã, o utilizador avança ou recua os cilindros pneumáticos da aparafusadora topo.
3. Ao pressionar no ecrã, o utilizador avança ou recua os cilindros pneumáticos da aparafusadora base.
4. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização da zona de amolgar da mesa de fixação.

Ao pressionar na área 'R2', o utilizador é direcionado para a janela da ferramenta do robô 2, Figura 3.39. Nesta, é possível ativar ou desativar o vácuo do *gripper*, bem como observar o estado do mesmo. Isso também se verifica quando o robô se encontra em *home* e em modo automático. Ao pressionar um botão, o utilizador regressa para a janela de estados da célula.

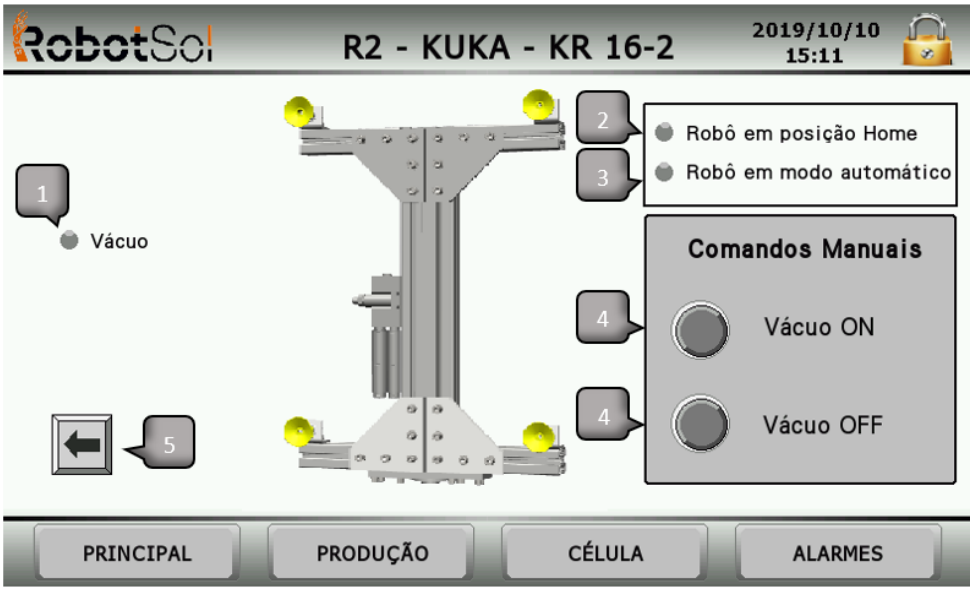


Figura 3.39: Ecrã correspondente à zona do transportador de saída

Legenda da Figura 3.39:

1. Indica o estado do vácuo (ativo torna-se verde, inativo cinzento).
2. Indica se o robô está em posição *Home* (ativo torna-se verde, inativo cinzento).
3. Indica se o robô está em modo automático (ativo torna-se verde, inativo cinzento).
4. Ao pressionar no ecrã, o utilizador ativa ou desativa o vácuo (ativo torna-se verde, inativo cinzento).
5. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de estados da célula.

Na Figura 3.40 é possível observar a informação remetente aos alarmes ativos. A mesma indica a sua ocorrência, a hora que foi verificada na consola e qual o nome do alarme ocorrido. De lado, possui setas para se poder movimentar na tabela, caso existam vários alarmes e um botão que permite abrir o histórico de alarmes.

Figura 3.40: Ecrã correspondente aos alarmes ativos

Legenda da Figura 3.40:

1. Tabela com os alarmes ativos. A tabela transmite informação sobre a data e hora de ocorrência, a hora de quando este foi verificado na consola e mensagem correspondente a cada alarme.
2. Ao pressionar no ecrã, o utilizador anda para cima e para baixo na tabela de visualização dos alarmes ativos.
3. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de histórico de alarmes.

Na Figura 3.41 é possível visualizarem-se informações remetentes a todos os alarmes ocorridos no sistema. As mesmas indicam o dia e hora da sua ocorrência, a hora que foi verificada na consola, o seu cancelamento e qual o nome do alarme ocorrido. De lado, possui setas para se poder movimentar na tabela, caso existam vários alarmes, um botão que permite limpar os registos da tabela e um botão que permite regressar à janela de alarmes ativos.

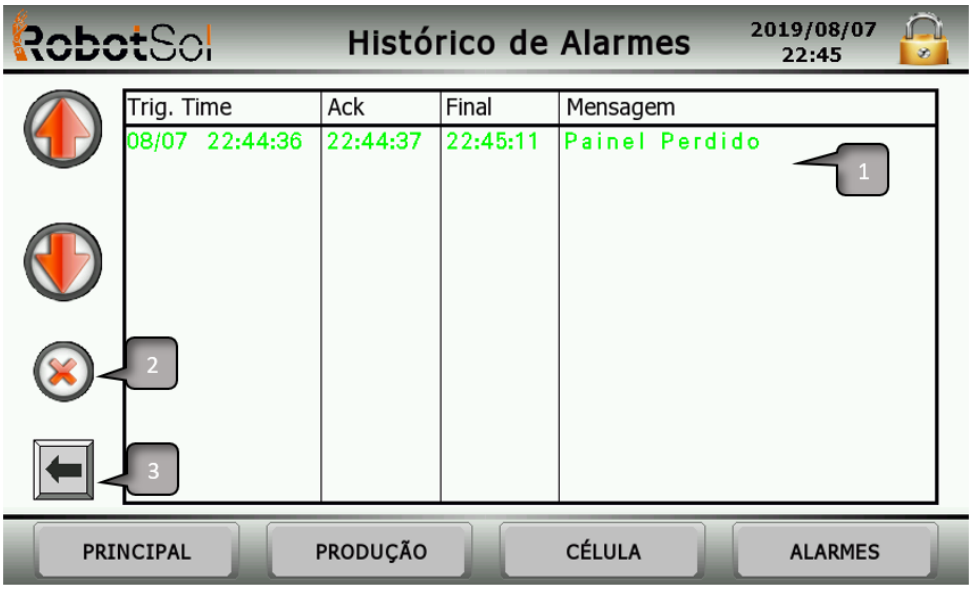


Figura 3.41: Ecrã correspondente ao histórico de alarmes

Legenda da Figura 3.41:

1. Tabela com os alarmes ativos. A tabela transmite informação sobre a data e hora de ocorrência, a hora de quando este foi verificado na consola, a hora do seu cancelamento e a mensagem correspondente a cada alarme.
2. Ao pressionar no ecrã, o utilizador limpa os registos dos alarmes que se encontram na tabela.
3. Ao pressionar no ecrã, o utilizador é direcionado para a janela de visualização de alarmes ativos.

Observe-se o apêndice I onde a Figura I.1 retrata a janela de contactos. Qualquer utilizador pode ver esta janela, mas, as opções que se encontram no lado direito (ver sinais dos robôs, produção e gestão de utilizadores), só são visíveis a utilizadores com permissão.

A Figura I.3 contêm informações da célula em forma de gráficos e tabelas.

Finalmente, observa-se a Figura I.4 e poder verificar-se um alarme da célula, correspondente à perda de um painel do robô 2.

3.10 Otimização da célula

Após a realização de testes e do desenvolvimento de códigos necessários ao funcionamento correto da célula, realizaram-se otimizações da mesma. Assim, neste subcapítulo, retrata-se a análise da solução otimizada, bem como as simulações realizadas. O programa utilizado para a simulação intitula-se de *kuka sim* 2.2.

3.10.1 Análise da solução otimizada

O funcionamento da solução otimizada envolve a utilização do robô 1 para colocar as peças de plástico e de alumínio na mesa de indexação. O robô 2 coloca os painéis na mesa de indexação e leva o quadro, montado, para a mesa de fixação. Assim, é necessário modificarem-se as ferramentas dos robôs (*gripper*) para a nova função. O *gripper* do robô 1 é modificado para conseguir pegar em duas peças ao mesmo tempo, enquanto que o *gripper* do robô 2 é alterado para conseguir pegar em dois quadros ao mesmo tempo.

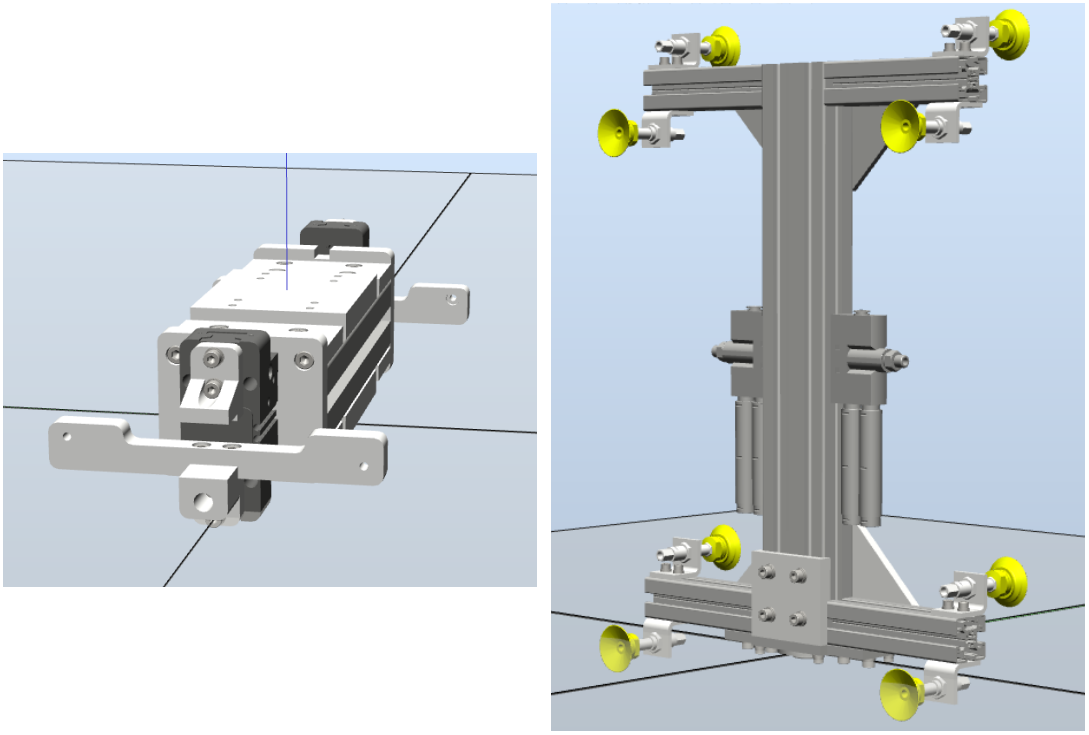
O funcionamento da célula tem uma nova sequência, que começa com o robô 2 a dirigir-se ao armazém de painéis. O mesmo, pega em dois painéis e, de seguida, coloca um na mesa de indexação. O robô 1 coloca as peças necessárias na mesa de indexação, para o mesmo ser indexado (pega em duas peças ao mesmo tempo). Seguidamente, o robô 2 pega no quadro completo e coloca o painel na mesa de indexação. Dirige-se para a mesa de fixação para deixar o quadro completo, enquanto que o robô 1 coloca as peças no painel deixado anteriormente. O robô 2, depois de colocar o quadro completo na mesa de fixação e enquanto o robô 1 coloca as peças, vai buscar um painel e espera. A partir desse momento, o ciclo repete-se.

Devido à nova sequência, o *layout* da célula foi alterado. Ver apêndice J.

3.10.2 Análise dos componentes de simulação

Antes de começar a simulação do novo *layout*, foi necessário rever os componentes da célula. Observaram-se os componentes utilizados e modificaram-se os necessários.

Os primeiros componentes que necessitaram de modificação, foram os *grippers*. Observando a Figura 3.42 verificam-se as alterações realizadas nos mesmos. O *gripper* 1 consegue pegar em duas peças ao mesmo tempo, quer sejam de plástico ou de metal, enquanto que o *gripper* 2 consegue pegar em dois quadros ou painéis ao mesmo tempo.



(a) Esquemático 3D *gripper* robô 1 **(b)** Esquemático 3D *gripper* robô 2

Figura 3.42: *Grippers* atualizados

No apêndice J, é possível observarem-se os restantes componentes modificados.

O tapete de entrada encontra-se com os acessórios de apoio na posição errada (necessário rodar 180º), bem como as peças de metal e de plástico, que necessitam estar com orientações opostas. De notar que não se atualizou o transportador de entrada completamente, mas sim, a parte necessária à simulação.

Devido à necessidade de simular a célula rapidamente, foi modificado o imprescindível para testar a mesma. Devido a isso, encontraram-se vários erros no esquemático 3D da mesa de indexação e fixação, mas como os mesmos não interferiam na simulação, não foram modificados.

3.10.3 Simulação Robô 1

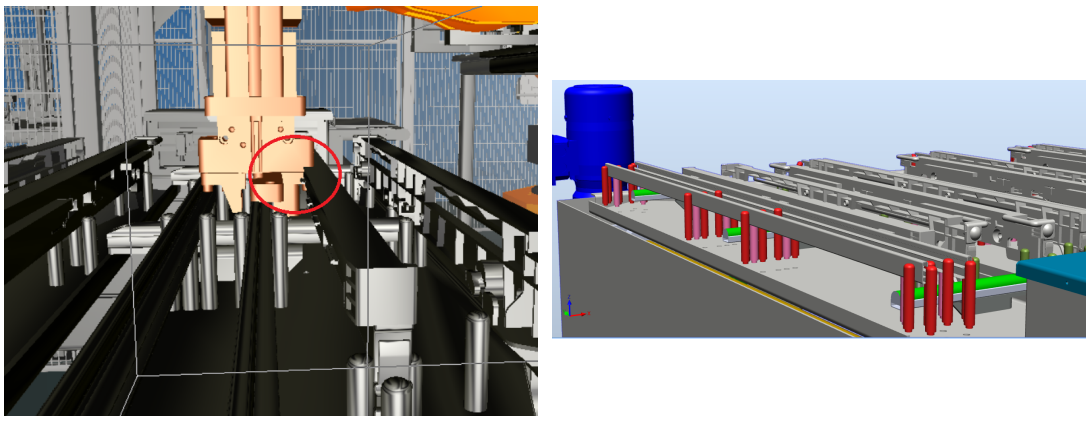
Neste subcapítulo, explicam-se as simulações realizadas com o robô 1, bem como as melhorias envolvidas.

Os aspetos que se têm de ter atenção remetem-se a dois momentos. O primeiro momento envolve o transportador de entrada, em que é necessário colocar o *gripper* na posição correta para agarrar as peças (*pick*). O segundo momento envolve a mesa de indexação, em que o *gripper* coloca as peças nas posições corretas (*place*). Assim, para o primeiro momento, pode observar-se na Figura 3.43a que

o *gripper* colide com a peça de plástico. Esta colisão só acontece ao pegar nas peças de metal. Para solucionar esse problema, colocaram-se todos os suportes com a mesma altura (em Z). Isso poderá confirmar-se na Figura 3.43b. Com esta alteração, o *pick* ficou parcialmente solucionado, devido ao facto de se ter encontrado outro problema, ou seja, dois parafusos colidiam com as peças envolventes.

No apêndice J, na Figura J.4, verifica-se a solução encontrada. Modificaram-se os parafusos de forma a não ficarem salientes e dentro do *gripper*. Assim, o *pick* ficou solucionado.

No segundo momento, não foi encontrado nenhum problema, sendo o trajeto percorrido sem percalços. No apêndice J, na Figura J.6 verifica-se o *place* das peças de metal e de plástico.



(a) Esquemático 3D da colisão do *gripper* robô 1

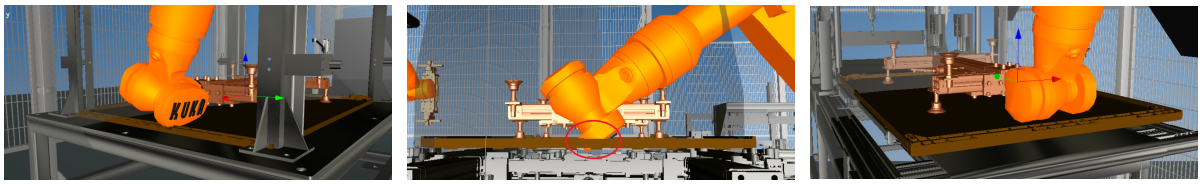
(b) Esquemático 3D do transportador de entrada atualizado

Figura 3.43: Solução encontrada para o tapete de entrada

3.10.4 Simulação Robô 2

Após a demonstração da simulação do robô 1, explicam-se as simulações realizadas com o robô 2, bem como as melhorias envolvidas.

Nesta parte da simulação é necessário ter em conta três situações. A primeira envolve o *pick* dos painéis no armazém de painéis, Figura 3.44a. A segunda remete-se ao *place* do painel e ao *pick* do quadro na mesa de indexação, Figura 3.44b. Por fim, a última situação engloba o *place* do quadro na mesa de fixação, Figura 3.44c.



(a) Esquemático 3D da posição *pick* no armazem de painéis

(b) Esquemático 3D da posição *pick*/*place* da mesa de indexação

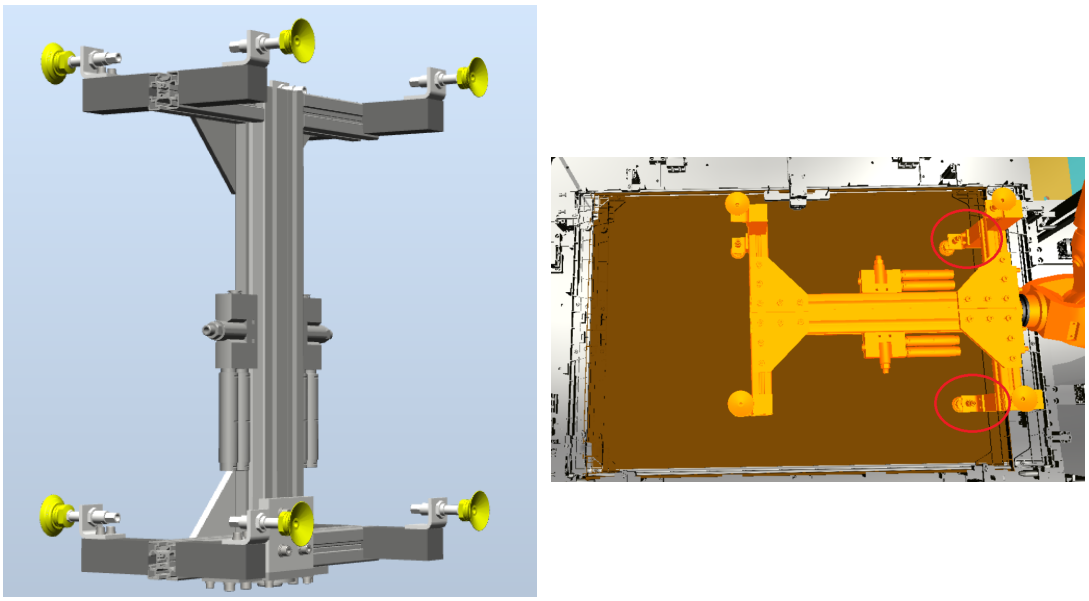
(c) Esquemático 3D da posição *place* da mesa de fixação

Figura 3.44: Esquemático 3D da primeira simulação do robô 2

Após uma primeira simulação, modificaram-se alguns aspetos relativos à célula. Assim, observando o apêndice J, Figura J.1, verifica-se o *layout* inicial otimizado.

A partir deste *layout*, modificaram-se vários componentes, nomeadamente: a mesa de indexação foi afastada 100mm para a direita da sua posição original; o robô 1, bem como o pedestal, foram centrados com a mesa de indexação; o robô 2 foi mudado de posição, devido ao *pick* na mesa de indexação ficar com 90° em relação ao *gripper*; o armazém de painéis foi centrado à beira da mesa de indexação.

Todas estas modificações não foram suficientes para o efeito pretendido, tendo havido necessidade de alterar o *gripper*. O mesmo pode ver-se na Figura 3.45a e sofreu três alterações. A primeira, envolve o desfasamento de 100mm das ventosas, nos dois lados. A segunda alteração, retrata a modificação da posição das ventosas, tornando o *gripper* simétrico. Por fim, a terceira alteração, tem em vista a necessidade de acrescentar uma ventosa, no meio do *gripper*, nos dois lados. Esta última alteração não é visível na simulação, pelo facto de não ser necessária para testes. Efetivamente, a ventosa ajuda no suporte do quadro, assim, é necessário acrescentá-la na realidade. Por fim, modificou-se a posição de *pick* na mesa de indexação, deixando de ser no centro do painel, mas sim encostado, Figura 3.45b.



(a) Esquemático 3D do *gripper* robô 2 final **(b)** Esquemático 3D do *pick* do armazém de painéis

Figura 3.45: Alterações realizadas na simulação do robô 2

No apêndice J encontram-se as restantes imagens referentes às simulações finais realizadas. O processo de simulação realizou-se, testando cada alteração descrita anteriormente, concluindo um sistema operacional, de acordo com a otimização pensada previamente.

3.10.5 Análise da solução otimizada final

Nem sempre a solução mais otimizada corresponde à solução real. Neste caso, a simulação demonstrada no robô 2 não pode ser realizada na realidade. O problema deve-se ao facto de que o peso do *gripper* excede a carga que o robô suporta. O quadro completo pesa 7Kg e o painel, sem as peças pesa 6Kg. A junção desses pesos com a estrutura de suporte das ventosas, excede a carga máxima suportada pelo robô. Assim, o robô 2 fica com o *gripper* inicial, adicionando-se, contudo, uma ventosa. As restantes alterações simuladas remetentes à célula foram, também, implementadas. O seu funcionamento envolve, a colocação do painel na mesa de indexação e o transporte do quadro montado para a mesa de fixação. O robô 1 coloca as peças de plástico e de alumínio na mesa de indexação (duas ao mesmo tempo).

Capítulo 4

Resultados Experimentais

Neste capítulo, estão descritos os testes realizados na célula. Os mesmos demonstram o progresso e melhorias do sistema ao longo das suas fases. Inicialmente, são abordados os testes para validação da mesa de fixação e indexação. De seguida, são demonstrados os tempos para a contagem do tempo de ciclo da solução funcional e da simulação otimizada e é apresentada uma comparação entre esses tempos.

4.1 Validação de componentes

No decorrer do projeto, realizaram-se testes na empresa *RobotSol* para corrigir o funcionamento do sistema. Neste projeto, os testes iniciaram-se com a validação de certos componentes do sistema, não sendo necessário validar as trajetórias dos robôs, pois estavam validadas e funcionais. Assim, testaram-se as mesas de indexação e de fixação. Com isto, após as ligações físicas realizadas e configurada a comunicação da célula, testaram-se e verificaram-se as entradas, saídas e a comunicação com os restantes componentes da célula. Atuaram-se as electroválvulas, visualizou-se o estado dos sensores e ativaram-se as ferramentas dos robôs. Ao realizar esta operação, foi possível visualizar o movimento dos cilindros pneumáticos, devido à atuação das electroválvulas. Verificaram-se, no robô, as alterações dos estados das respetivas variáveis que se alteravam no *PLC* e vice-versa. Por fim, verificou-se o funcionamento da consola *HMI*. Após a sua configuração, validou-se a mesma, trocando sinais entre a consola e o *PLC*. Como referido no capítulo anterior, ao pressionar os botões da consola *HMI*, é possível observar o movimento dos cilindros pneumáticos e verificar o estado dos sensores. Também é possível observarem-se os sinais no *PLC*, verificando se o mesmo chegava ao *PLC*, ou não.

4.1.1 Testes na mesa de indexação

Como referido no capítulo anterior, realizaram-se testes na mesa de indexação, que serviram como forma de aprendizagem, e algumas soluções encontradas sofreram alterações e melhorias. Assim, realizaram-se testes com 16 painéis, sendo que se selecionaram 32 peças laterais, nomeadamente, 16 peças metálicas e 16 peças de plástico.

- Divide-se a classificação dos testes como:

  Positivo: As peças encaixam, totalmente, no painel.

  Negativo: As peças encaixam, parcialmente, no painel.

  Falha: Existem muitas falhas nos encaixes.

- Verificam-se os seguintes resultados:

  Positivo: 7/16 = 43,75%

  Negativo: 7/16 = 43,75%

  Falha: 2/16 = 12,50%

Ao observarem-se os resultados, confirma-se que o erro não é admissível (>56,25%). Cerca de sete peças encaixaram totalmente, sendo que, as restantes não estavam totalmente encaixadas, ou tinham muitas falhas. Como referido no capítulo anterior, a mesa de indexação sofreu alterações e melhorias, tornando o erro menor. Os testes preliminares executados não têm testes de comparação, pois a mesa de indexação não faz parte dos objetivos iniciais propostos.

4.1.2 Testes na mesa de fixação

Devido à falta de material disponível na empresa *RobotSol*, realizaram-se testes, aproveitando-se ao máximo o material existente. Os mesmos foram realizados em pequenos grupos, retirando-se o máximo de conclusões possíveis. Assim, realizaram-se dois tipos de testes. O primeiro teste retrata a utilização, ou não, das bases das aparafusadoras e o segundo remete-se ao binário das aparafusadoras.

No primeiro teste, utilizaram-se 10 quadros montados, no entanto, estes não estavam totalmente imaculados, pois continham furos preexistentes. Os quadros furados, foram distribuídos igualmente pelos dois testes, existindo um para cada teste. O binário de cada aparafusadora encontrava-se no mínimo.

- Dividiu-se a classificação dos testes como:

  Positivo: Os quadros são, totalmente, aparafusados à frente e a trás.

  Negativo: Os quadros não são aparafusados à frente e/ou a trás.

- Para o primeiro teste (base das aparafusadoras recuadas), verificaram-se os seguintes resultados:

  Positivo: 0/5 = 0%

  Negativo: 5/5 = 100%

- Para o segundo teste (base das aparafusadoras avançadas), verificaram-se os seguintes resultados:

  Positivo: 3/5 = 60%

  Negativo: 2/5 = 40%

Para os testes serem classificados com negativos, basta que algum dos quatro furos não seja realizado com sucesso. Assim, observa-se o apêndice K, Figura K.1 e a Figura K.2, e verifica-se que os testes realizados provam que a base da aparafusadora avançada tem melhores resultados.

No segundo teste, realizaram-se uma série de testes com valores diferentes de binário. Como mencionado no capítulo três, cortaram-se as laterais das peças de plástico para se realizarem testes ao longo do metal (que não tem furos). Assim, foi possível testar a força necessária para furar o metal e a madeira, confirmando, ou não, se a aparafusadora servia para o objetivo proposto.

Realizaram-se quatro testes diferentes e em cada um furou-se cinco vezes o quadro completo. Ao analisar-se o teste anterior, é possível concluir que é necessário aumentar o binário das aparafusadoras. Assim, começou-se com o binário das aparafusadoras no nível 4 (13 voltas).

- Dividiu-se a classificação dos testes como:

  Positivo: Os quadros são, totalmente, aparafusados do lado direito e do lado esquerdo.

  Negativo: Os quadros não são aparafusados do lado direito e/ou do lado esquerdo.

  Lado esquerdo: Os quadros são, totalmente, aparafusados no lado esquerdo.

  Lado direito: Os quadros são, totalmente, aparafusados no lado direito.

- Para o primeiro teste (binário das aparafusadoras no nível 4 (13 voltas)), verificaram-se os seguintes resultados:

  Positivo: 3/5 = 60%

  Negativo: 2/5 = 40%

  Lado esquerdo: 5/5 = 100%

  Lado direito: 3/5 = 60%

Conclui-se que é necessário aumentar o binário do lado direito. Realizou-se um teste com o binário da aparafusadora direita, com mais 15 voltas do que o anterior.

- Para o segundo teste (binário da aparafusadora direita, com mais 15 voltas do que o anterior), verificaram-se os seguintes resultados:

  Positivo: 3/5 = 60%

  Negativo: 2/5 = 40%

  Lado esquerdo: 3/5 = 60%

  Lado direito: 5/5 = 100%

Conclui-se que é necessário aumentar o binário do lado esquerdo. Realizou-se um teste com o binário da aparafusadora esquerda, com mais 15 voltas do que o anterior.

- Para o segundo teste (binário da aparafusadora esquerda, com mais 15 voltas do que o anterior), verificaram-se os seguintes resultados:

  Positivo: 1/5 = 20%

  Negativo: 4/5 = 80%

  Lado esquerdo: 1/5 = 20%

  Lado direito: 5/5 = 100%

Conclui-se que é necessário aumentar o binário do lado esquerdo. Realizou-se um teste com o binário da aparafusadora esquerda, com mais 20 voltas do que o anterior.

- Para o segundo teste (binário da aparafusadora esquerda, com mais 20 voltas do que o anterior), verificaram-se os seguintes resultados:

  Positivo: 5/5 = 100%

  Negativo: 0/5 = 0%

  Lado esquerdo: 5/5 = 100%

  Lado direito: 5/5 = 100%

Com os testes realizados, conclui-se que as aparafusadoras conseguem perfurar o metal e a madeira e que o aparafusamento não é constante. Estes testes podem ser observados no apêndice K e foram realizados antes da verificação do funcionamento das aparafusadoras, sendo que, após o seu melhoramento, estas têm um funcionamento constante.

Não foi possível realizarem-se testes finais na mesa de indexação, devido a dois motivos. O primeiro, remete-se à falta de material e o segundo, deve-se ao facto de que os *stoppers* eram peças provisórias. Estes, serviam para realizar pequenos testes, mas para séries de testes, os mesmos perdiam a posição provocando erros na célula. Desta forma, foi decidido pela empresa *RobotSol*, que se iria verificar o erro associado a este componente, realizando-se uma série de testes no cliente. A mesa de fixação encontra-se operacional nas pequenas quantidades de testes realizados.

4.2 Testes de tempo de ciclo

Nesta secção, são abordados os tempos de ciclo obtidos na primeira solução funcional e na simulação final. É, também, feita a análise e comparação, entre si, desses tempos.

4.2.1 Tempo real - Primeira solução

Após o sistema ficar funcional e cada mesa estar operacional, foram analisados os tempos de ciclo do sistema. Como referido anteriormente, só existia uma ficha 'x11' para os dois controladores. Por isso, foram retirados os tempos de cada robô, individualmente.

Na Figura 4.1, observa-se um tempo de ciclo de 128 s. É, ainda, possível verificar que o robô 1 utiliza 96 s na colocação do quadro e das respetivas peças, enquanto que o robô 2, demora, apenas, 14s no seu movimento. Uma otimização realizada nesta primeira solução, remete-se à montagem do quadro. Esta, inicia-se quando existe um painel e as duas peças de metal na mesa de indexação, não sendo necessário esperar pela colocação de todas as peças.

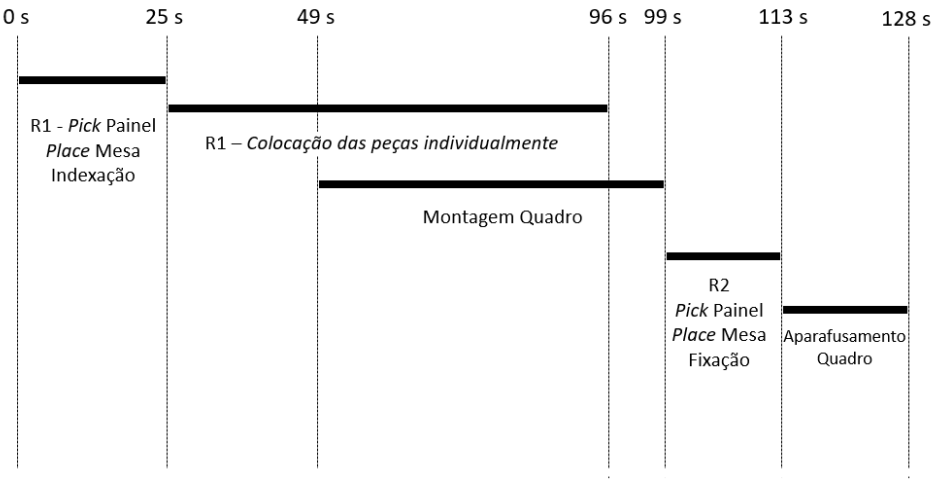


Figura 4.1: Diagrama de tempo de ciclo da célula com a primeira solução

4.2.2 Tempo simulação - otimizado

Retiraram-se os tempos de ciclo da célula, através da utilização do software de simulação (*kuka sim* 2.2.). Com esta, é possível obter o tempo total que cada robô demora a realizar as suas rotinas. Este tempo é a soma de dois tempos, nomeadamente, o tempo de movimento e o tempo de espera. Quando uma mesa necessita realizar uma tarefa, por exemplo, a mesa de indexação monta duas peças, utilizam-se comandos especiais ('*WAIT*'). O robô não pode estar presente na área da mesa, enquanto a mesma está em funcionamento. Por isso, param-se os robôs, simulando assim o tempo gasto da mesa. O mesmo acontece quando um robô entra numa área e o outro tem que esperar para que o primeiro saia dessa área. A simulação não engloba tempos da transferência de informação entre o robô e o *PLC*, ou *PLC* e as mesas.

Na Figura 4.2, observa-se um tempo de ciclo de 94s. Verifica-se que o robô 1 utiliza 37s na colocação

das peças. O robô 2, demora 25s na colocação do painel na mesa de indexação e 14s na colocação do quadro completo na mesa de fixação, tendo um total de 39s de funcionamento. A montagem do quadro começa quando existem peças suficientes.

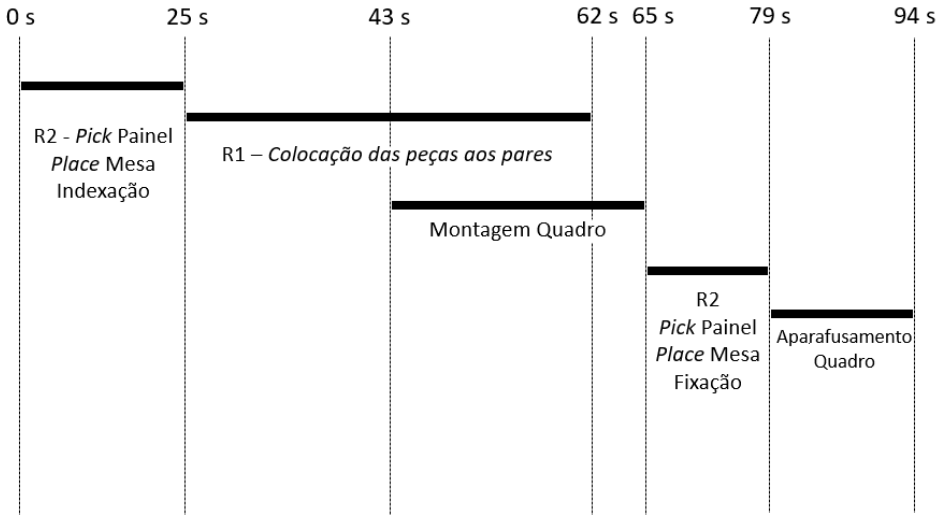


Figura 4.2: Diagrama de tempo de ciclo da célula com a solução otimizada

4.2.3 Comparação de tempos de ciclo

Obtiveram-se os tempos de ciclo em diferentes fases do sistema e compararam-se esses valores, Figura 4.3. Como observado nos diagramas de tempo de ciclo anteriores, a primeira solução tem um tempo de 128s e a solução otimizada tem um tempo de 94s. Isto, representa uma redução de 26.5% no tempo de ciclo da primeira solução para o tempo de simulação. Verifica-se que a solução encontrada na otimização, é distribuir as tarefas pelos dois robôs. O robô 1 fica encarregue das peças, conseguindo transportar duas peças ao mesmo tempo. O robô 2 transporta o painel e o quadro completo. Esta junção, levou à redução referida.

A nível da otimização realizada na primeira solução envolveu, também, a correção de peças da mesa de indexação. As peças corrigidas foram os *stoppers* e as bases da aparafusadora. A necessidade destas alterações deveu-se à calibração do ponto de aparafusamento, isto é, a posição e a orientação que o quadro tem de ter para ser aparafusado corretamente. Assim, os *stoppers* controlavam a distância entre os mesmos e a aparafusadora. Em relação à primeira solução, existia apenas um *stopper* que provocava erros na distância e na orientação do quadro. Com esta mudança, conseguiu-se corrigir totalmente o erro de distância mas o erro associado à orientação encontrava-se parcialmente corrigida, para tal seria necessário a segunda mudança. Como referido no capitulo anterior, as bases das aparafusadoras continham erros. As mesmas não tocavam no quadro e encontravam-se desniveladas. Na solução otimizada as

bases da aparafusadora foram modificadas para uma peça em forma de 'L' e niveladas. Desta maneira, o quadro encaixava-se na peça, ficando com a orientação desejada. A junção destas correções provocou a correção no aparafusamento dos quadros. Por fim, na zona de amolgar não foi necessário mudar os componentes, apenas utilizou-se uma ponteira para obter o resultado pretendido.

Uma futura redução do tempo de ciclo envolve a certeza da posição do *pick* do painel. Esta solução visa um suporte em que o robô 2 coloque o painel, e fique preso numa posição conhecida. A mesa de indexação, não necessita de centrar o quadro, apenas de o segurar. Assim, o robô 2 pega no quadro, na posição correta. A sequência da célula começa pela colocação das duas primeiras peças na mesa de indexação, pelo robô 1. O robô 2, pega no quadro que se encontra no suporte e posiciona-o na mesa de indexação. Enquanto isso acontece, o robô 1 pega nas restantes peças. Quando o painel se encontra na mesa de indexação, a mesa monta as duas primeiras peças e o painel. De seguida, o robô 1 coloca as duas peças em posição de montagem e a mesa monta o restante quadro. O robô 2 retira o quadro e leva-o para a mesa de aparafusamento, repetindo o ciclo. Ao fazer esta pequena mudança, torna-se possível reduzir o tempo de ciclo, conseguindo assim satisfazer a exigência de produção do cliente.

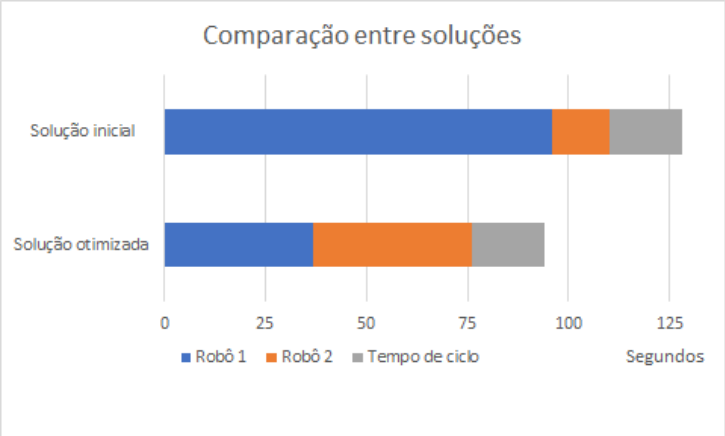


Figura 4.3: Comparação entre tempos de ciclo

Capítulo 5

Conclusão

A dissertação apresentada permitiu retificar um sistema robotizado de montagem automático para substituição de linhas manuais tradicionais. Esta linha de montagem automática encontrava-se disfuncional e os principais problemas situavam-se nas mesas de indexação e fixação. Após uma série de ações, nomeadamente, o estudo realizado, a análise da primeira solução, os testes e melhorias realizados nas mesas, o desenvolvimento dos códigos do robô, do *PLC* e do *HMI* e o desenvolvimento da simulação para otimização da célula, pode-se concluir que o principal objetivo proposto foi, parcialmente, cumprido, uma vez que, a célula se encontra funcional, mas não cumpre a produção exigida (2 a 2.5 quadros/min). Isto é, todos os componentes estão funcionais (mesa de fixação, mesa de indexação, armazém de painéis, transportador de entrada e de saída) e o objetivo de montar um quadro completo está cumprido, no entanto, o tempo de ciclo é superior ao desejado. Assim, para cumprir na totalidade o objetivo proposto, é necessário executar a melhoria final mencionada no capítulo anterior. Com isto, conclui-se que houve um melhoramento do tempo de ciclo (34s) desde a primeira solução até à solução final. É necessário ter em conta que os objetivos secundários de implementação da solução no cliente e afinações finais associadas, não foram realizadas, devido ao atraso da entrega de material na empresa *RobotSol*.

Antes de se iniciar o projeto, foi realizado um estudo teórico relativo à robótica industrial e fez-se uma breve introdução a conceitos importantes na robótica. Verificaram-se casos de estudo de redução de tempo de ciclo, utilizando robôs, fez-se uma abordagem histórica da robótica e foram descritos os dispositivos que compõem uma célula robotizada e a sua função. Também se referiram características de desempenho de um robô industrial, explicaram-se quais os robôs utilizados e as suas configurações. Realizou-se, depois, uma pesquisa relativa à rede de comunicação industrial utilizada, sendo um componente essencial na automatização de processos. Nesta parte, foi ainda estudada a segurança relativamente à célula e aos robôs existentes. Após isto, explicou-se o funcionamento de um *PLC*, isto é, dos seus componentes, e efetuou-se um estudo direcionado à programação dos *PLCs*, descrevendo a linguagem utilizada, bem como instruções importantes. Por fim, descreveu-se a funcionalidade de um *HMI*. Para finalizar o estudo teórico, efetuou-se um estudo direcionado à programação de robôs. Este estudo teórico realizado antes do desenvolvimento do projeto, contribuiu para adquirir conhecimento, não só para a realização desta dissertação, mas também para trabalhos futuros direcionados para a área da robótica industrial. Após o estudo teórico, iniciou-se o desenvolvimento do projeto, começando-se pela análise

da primeira solução. Nesta análise, é visualizado o primeiro *layout* da célula e é descrito e estudado cada componente utilizado no sistema, com o objetivo de saber como cada equipamento funciona. De seguida, foi demonstrada a arquitetura do sistema, o tipo de comunicação efetuada entre os dispositivos e as ligações efetuadas na célula. Efetuaram-se melhorias nas mesas de indexação e fixação, tornando-as funcionais. Depois, através de fluxogramas, descreveu-se o código desenvolvido para o robô 2, o *PLC* e a *HMI*. Por fim, criou-se o *layout* na simulação, otimizando o máximo possível a célula.

No capítulo seguinte, foram apresentados os testes e resultados do sistema. É abordado o tempo de ciclo do sistema e feita uma comparação da primeira solução com a solução otimizada. Observou-se uma redução significativa no tempo de ciclo que permite realizar um aumento na produção. Por fim, explicou-se uma possível otimização da célula que cumpriria com o objetivo do cliente.

5.0.1 Trabalhos futuros

Os objetivos do sistema proposto nesta dissertação foram parcialmente cumpridos com sucesso. Para se concluírem esses objetivos, será necessário implementar a solução referida anteriormente, isto é, criar-se uma estrutura para apoio do painel que obrigará que a posição de *pick* do robô 2 seja a correta. Seria, também, interessante implementar a célula no cliente, existindo assim uma oportunidade de aprendizagem. No entanto, nenhum sistema é perfeito, existindo sempre oportunidade para melhoria contínua. Deste modo, poder-se-ia otimizar o código existente, bem como simplificar a programação efetuada no *HMI*.

Bibliografia

[1] A. Sousa, "Automatização de linha de montagem na Paulo Mendes S.A.," Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2009.

[2] P. Tavares, "Planeamento de trajetórias em Manipuladores em ambientes industriais," Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2015.

[3] J. Silva, "Recolha Robotizada de Rolhas," Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2012.

[4] A. Grau, M. Indri, L. Lo Bello, and T. Sauter, "Industrial robotics in factory automation: From the early stage to the Internet of Things," *Proceedings IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, vol. 2017-Janua, pp. 6159–6164, 2017.

[5] A. F. Ribeiro, "Robôs de serviços," *UNAGUI - Universidade Autodidacta de Guimarães*, 1998.

[6] Baxter, R. and Hastings, N. and Law, A. and Glass, E. J.., *Handbook of Robotics*, vol. 39. Springer, 2008.

[7] "Robots double worldwide by 2020 - international federation of robotics," June 2019. Disponível em https://ifr.org/ifr-press-releases/news/robots-double-worldwide-by-2020.

[8] J. P. Womack, D. T. Jones, and D. Roos, "A máquina que mudou o mundo," *World*, pp. 1–11, 1992.

[9] S. Y. Nof, *Handbook of Industrial Robotics*. 2 ed., 2008.

[10] Vitor Ferreira Romano, *Robótica Industrial: Aplicação na Indústria de Manufatura e de Processos*. Edgard Blucher, 1 ed., 2002.

[11] "Controlador KUKA e Teach Pendant." Disponível em `https://www.eplan.com.br/typo3temp/_processed_/8/5/csm_4416_661f7febfa.jpg` Acedido: 19-06-2019.

[12] V. Carrara, "Introdução à robótica industrial," Setembro 2018. Instituto Nacional de Pesquisas Espaciais. Disponível em `http://urlib.net/8JMKD3MGP3W34P/3K5JPL8`.

[13] "ISO 21748:2017(en), Guidance for the use of repeatability, reproducibility and trueness estimates in measurement uncertainty evaluation." SOURCE: ISO 3534 2:2006, 3.3.6, Disponível em `https://www.iso.org/obp/ui/#iso:std:iso:21748:ed-2:v1:en` Acedido: 13-06-2019.

[14] "ISO 21748:2017(en), Guidance for the use of repeatability, reproducibility and trueness estimates in measurement uncertainty evaluation." SOURCE: ISO 3534 2:2006, 3.3.3, Disponível em https://www.iso.org/obp/ui/#iso:std:iso:21748:ed-2:v1:en Acedido: 13-06-2019.

[15] "ISO 21748:2017(en), Guidance for the use of repeatability, reproducibility and trueness estimates in measurement uncertainty evaluation." SOURCE: ISO 3534-1, 3.3.3, Disponível em https://www.iso.org/obp/ui/#iso:std:iso:21748:ed-2:v1:en Acedido: 13-06-2019.

[16] "ISO 21748:2017(en), Guidance for the use of repeatability, reproducibility and trueness estimates in measurement uncertainty evaluation." SOURCE: ISO 5725-1:1994, 3.3.3, Disponível em https://www.iso.org/obp/ui/#iso:std:iso:21748:ed-2:v1:en Acedido: 13-06-2019.

[17] "KUKA KR 16." [ONLINE] Disponível em https://sealing-system.dk/en/packaging-solutions/kuka-kr-16/ Acedido: 16-01-2019.

[18] *Programação do robô 1,KUKA System Software 8, Documento de treinamento*. KUKA Roboter GmbH, 24.09.2012 ed., 2012.

[19] R. Zurawski, *Industrial Communication Technology Handbook*. CRC Press, 2 ed., 2015.

[20] "Robotics News - New Robot Safety Standard." [ONLINE] Disponível em https://www.robotics.org/content-detail.cfm/Industrial-Robotics-News/New-Robot-Safety-Standard/content_id/4133 Acedido: 2019-06-21.

[21] P. Davison, "Safety Standards and Collaborative Robots." [ONLINE] Disponível em https://www.robotics.org/userAssets/riaUploads/file/6-Pat.pdf Acedido: 2019-06-21.

[22] "DIRECTIVA 2006/42/CE DO PARLAMENTO EUROPEU E DO CONSELHO." [ONLINE] Disponível em https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2006:157:0024:0086:pt:PDF Acedido: 2019-06-21.

[23] "MINISTÉRIO DA ECONOMIA E DA INOVAÇÃO Decreto-Lei n.º 103/2008 de 24 de Junho." [ONLINE] Disponível em https://dre.pt/application/dir/pdf1s/2008/06/12000/0376503795.pdf Acedido: 2019-06-21.

[24] M. Silva, *Segurança na Programação de Operação de Robôs Industriais*.

[25] S. Chitra and V. Raghavan, "Conveyor Control Using Programmable Logic Controller," *International Journal of Advancements in Research & Technology*, vol. 3, no. 8, pp. 25–31, 2014.

[26] W.Bolton, *Programmable Logic Controllers*. Elsevier, 5 ed., 2011.

[27] J. Augusto and F. Ferreira, "Virtualização de autómatos programáveis," Master's thesis, Universidade de Aveiro, 1994.

[28] I. M. Parede and L. Gomes, *Automação Industrial Eletrônica*. Centro Paula Souza, 2013.

[29] "IEC 61131:2019 SER | IEC Webstore." [ONLINE] Disponível em `https://webstore.iec.ch/publication/62427` Acedido: 16-02-2019.

[30] K. Kamel and E. Kamel, *Programmable Logic Controllers - Industrial Control*. McGraw-Hill Education, 2014.

[31] "The Making of an Off Timer." [ONLINE] Disponível em `http://www.plcdev.com/making_an_off_timer_from_an_on_timer` Acedido: 2019-06-18.

[32] "5. Fundamentos da Programação LADDER," tech. rep. [ONLINE] Disponível em `http://www.lcvdata.com/eletrotecnica/ladder/Aula16-EPD030{_}Ladder1.pdf` Acedido: 2019-06-01.

[33] "Programmable Logic Controllers - PLC Timer Functions." [ONLINE] Disponível em `http://hvacelectricalinterviewtips.blogspot.com/2017/09/programmable-logic-controllers-plc.html` Acedido: 2019-06-18.

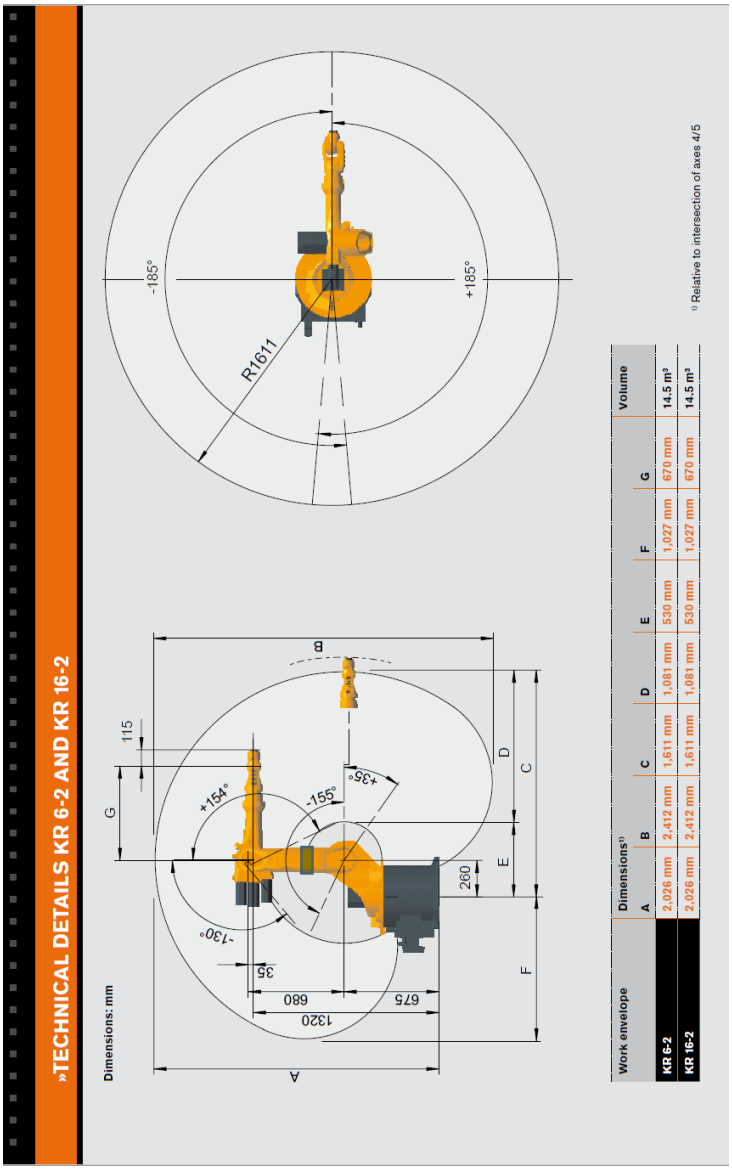
[34] F. Petruzella, *Programmable Logic Controllers*. Education, McGraw Hill, fifth edit ed., 2017.

[35] "Quadros Eléctricos–Definições e Normas ABB." [ONLINE] Disponível em `http://www.abb.pt/cawp/seitp202/96c64772df3a1910c1257578004f99c5.aspx` Acedido: 2019-07-03.

[36] KUKA Roboter GmbH, *KUKA System Software 8 . 3 Operating and Programming Instructions for System Integrators*. 14.01.2015.

Anexo A

KUKA - KR 16-2



Type	KR 6-2	KR 16-2
Maximum reach	1,611 mm	1,611 mm
Rated payload	6 kg	16 kg
Suppl. load, arm/link arm/rotating col.	10/variable/20 kg	
Suppl. load, arm + link arm, max.	Variable	
Maximum total load	36 kg	46 kg
Number of axes	6	
Mounting position	Floor, wall, ceiling	
Variant		Cleanroom, Foundry, Explosion-Proof
Positioning repeatability*	±0.05 mm	
Path repeatability*		
Controller	KR C2 edition2005	
Weight (excluding controller), approx.	235 kg	235 kg
Temperature during operation	+5 °C to +55 °C	
Protection classification	IP 65	
Robot footprint	500 mm x 500 mm	
Connection	7.3 kVA	
Noise level	< 75 dB	

Axis data	Range (software)	Speed with rated payload	
		6 kg	16 kg
Axis 1 (A1)	±185°	156°/s	156°/s
Axis 2 (A2)	+35°/–155°	156°/s	156°/s
Axis 3 (A3)	+154°/–130°	156°/s	156°/s
Axis 4 (A4)	±350°	343°/s	330°/s
Axis 5 (A5)	±130°	362°/s	330°/s
Axis 6 (A6)	±350°	659°/s	615°/s

Drive system, electromechanical with brushless AC servomotors

*to ISO 9283

Details provided about the properties and usability of the products are purely for information purposes and do not constitute a guarantee of these characteristics. The extent of goods delivered and services performed is determined by the subject matter of the specific contract. No liability accepted for errors or omissions.

Anexo B

KUKA - Sinais

6.17.4 Signal diagrams

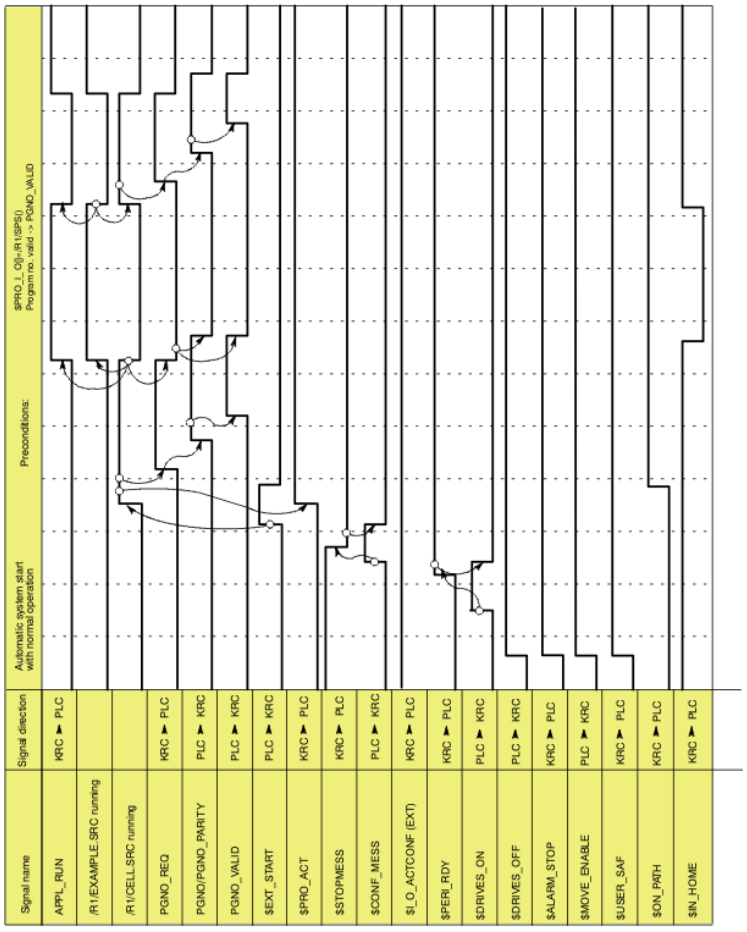


Fig. 6-23: Automatic system start and normal operation with program number acknowledgement by means of PGNO_VALID

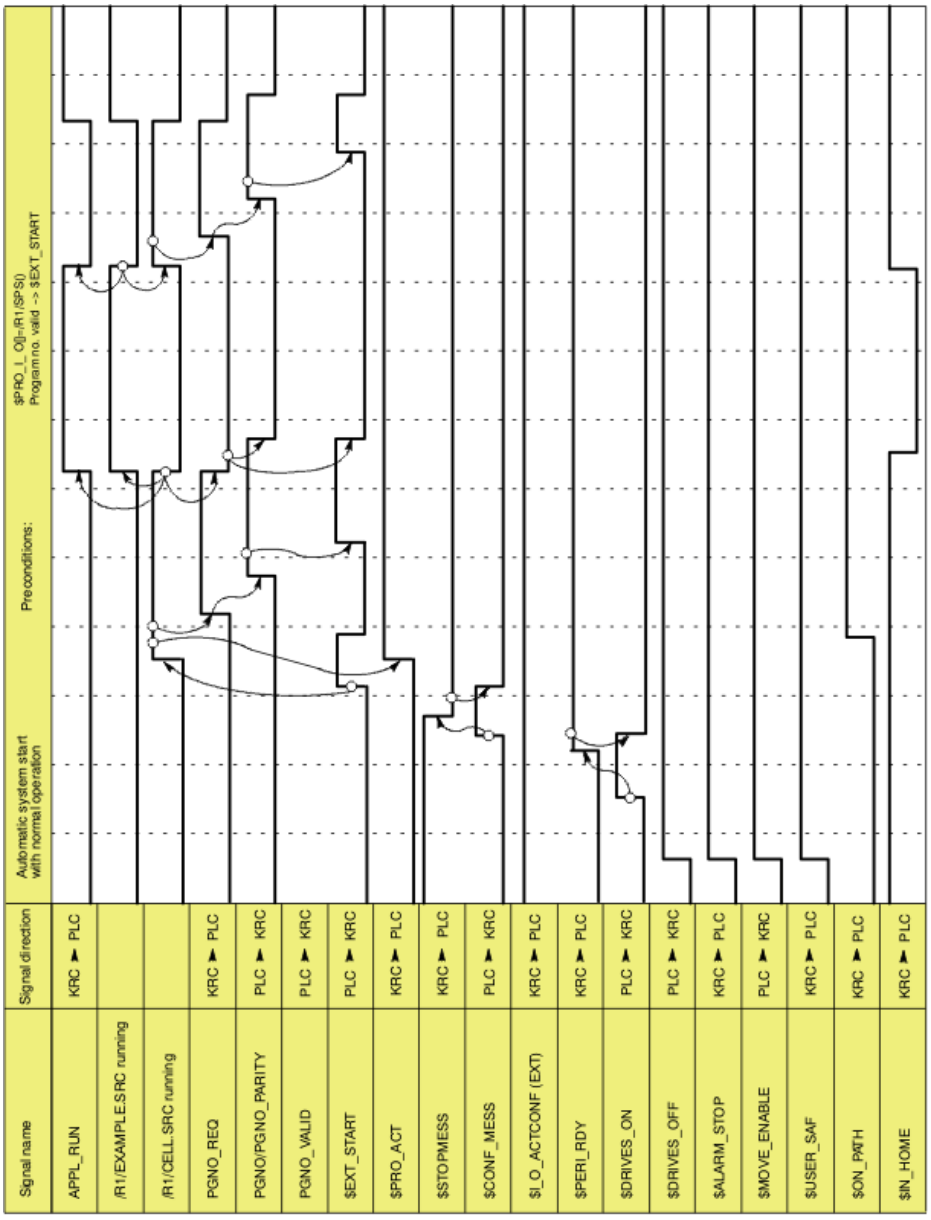


Fig. 6-24: Automatic system start and normal operation with program number acknowledgement by means of $EXT_START

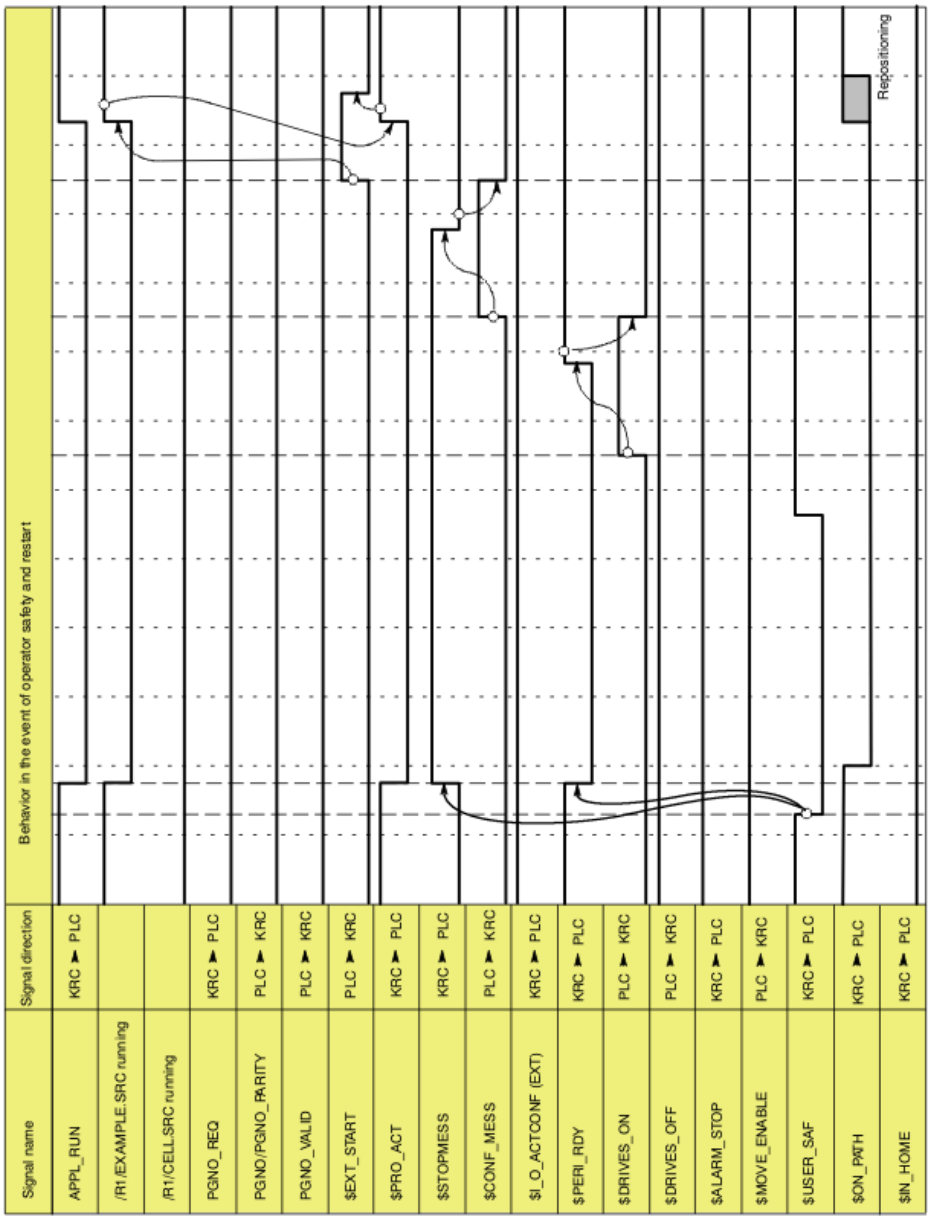


Fig. 6-25: Restart after dynamic braking (operator safety and restart)

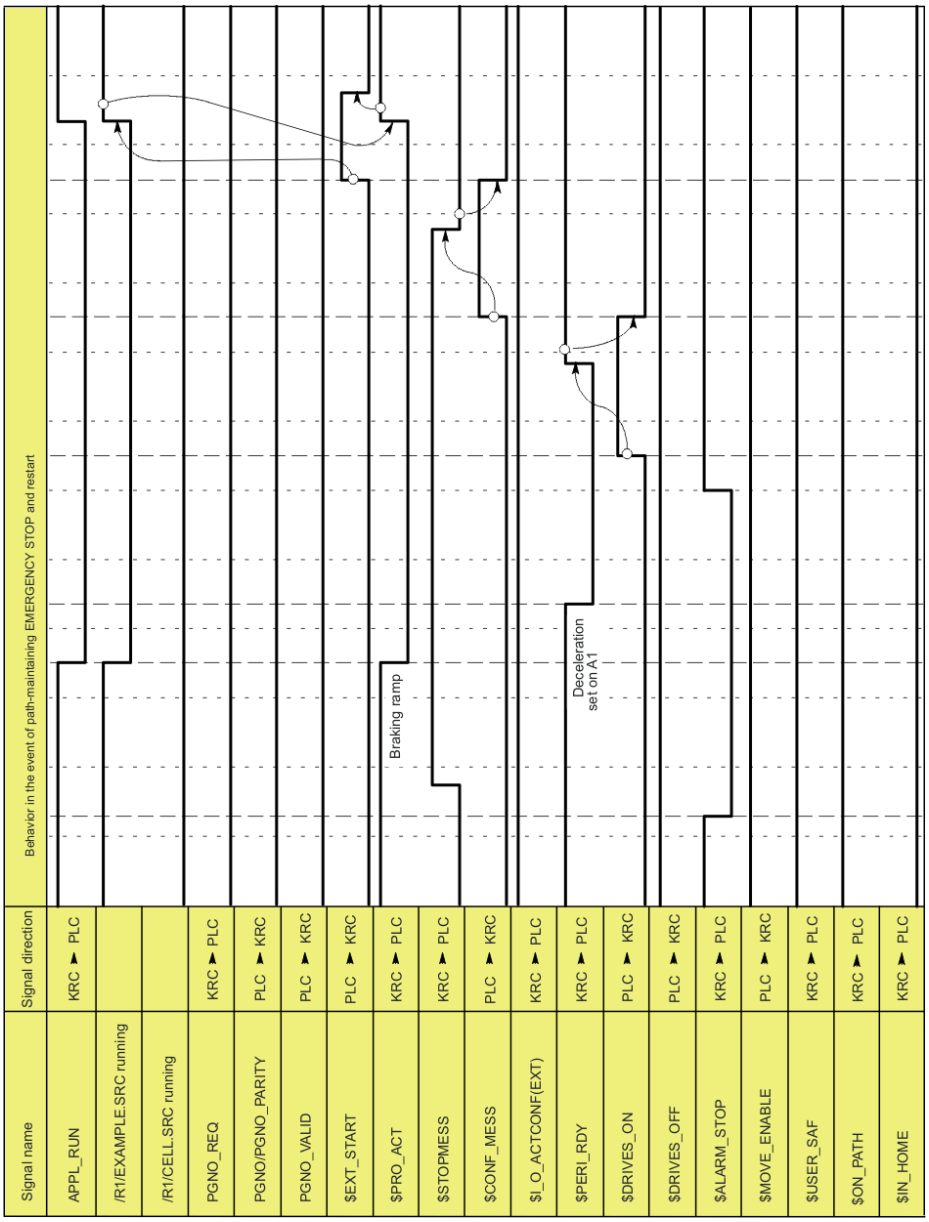


Fig. 6-26: Restart after path-maintaining EMERGENCY STOP

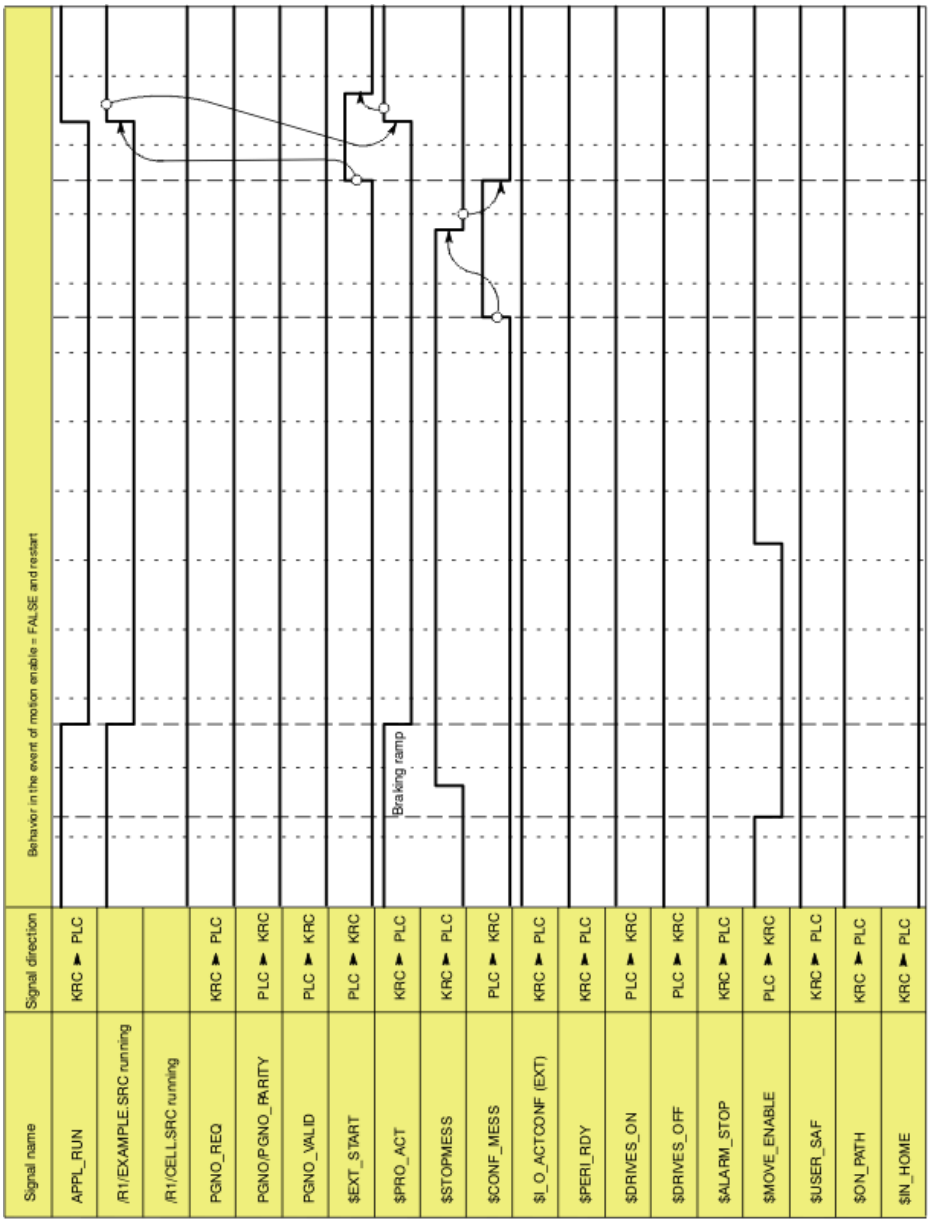


Fig. 6-27: Restart after motion enable

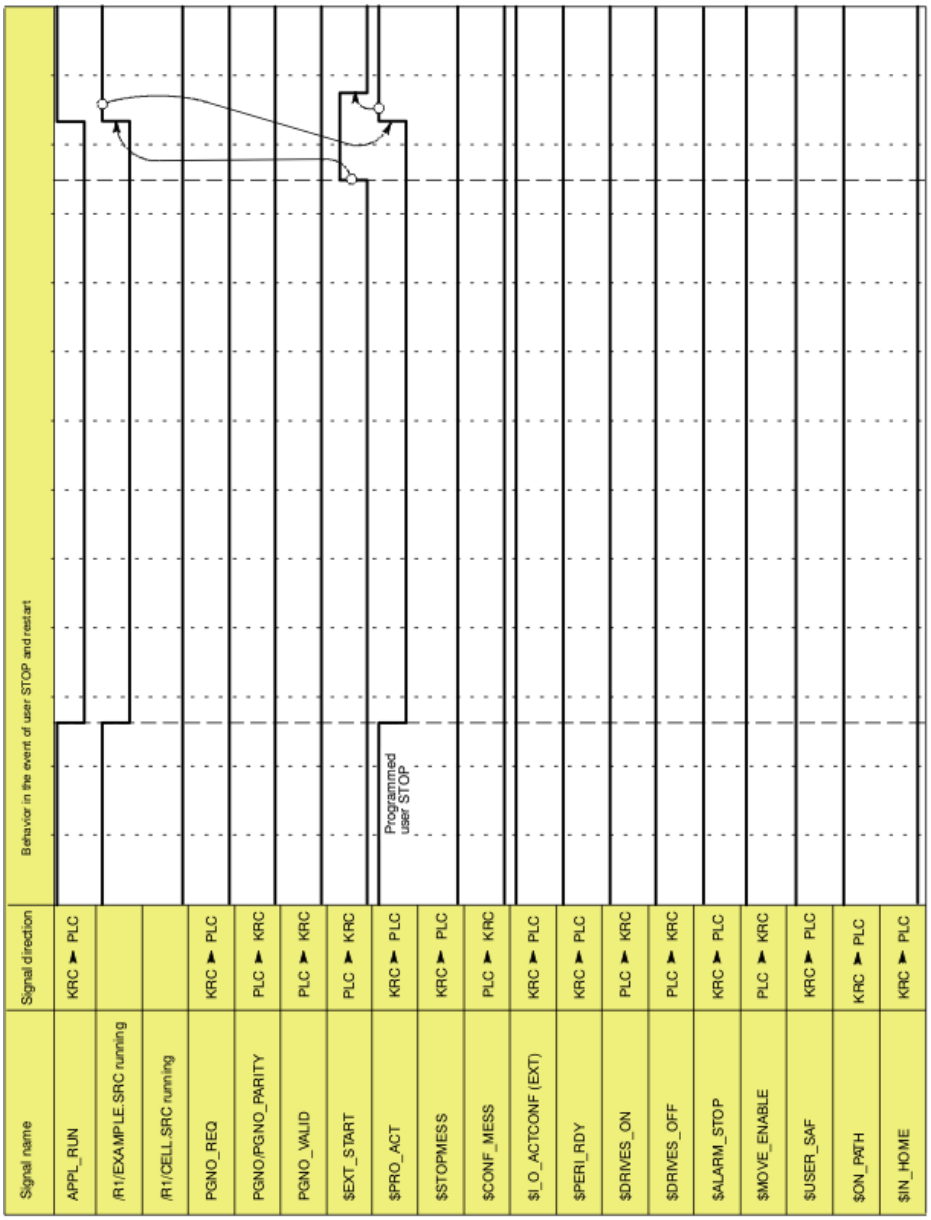


Fig. 6-28: Restart after user STOP

Anexo C

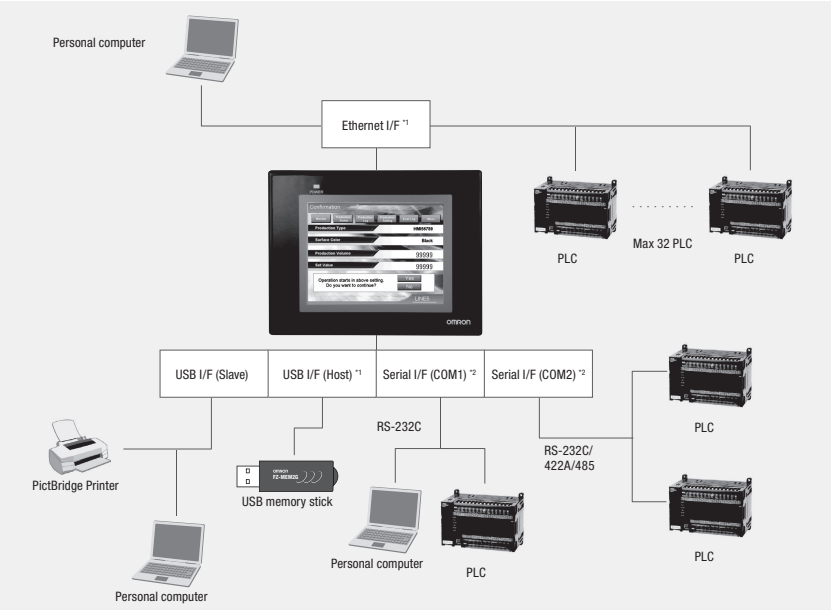
HMI NB Series

NB series

The feature-rich, economic HMI

- More than 65,000 display colours TFT touch screen
- Available in sizes ranging from 3.5 to 10 inches
- Long-life LED backlight
- Serial, USB or Ethernet communication
- USB memory stick support
- 128 MB internal memory
- Vector and bitmap graphics

System Configuration



*1. The Ethernet communication and USB Host are supported only by NB__-TW01.
*2. The NB3Q has only one port (RS-232C/422A/485) COM1.

NB series

Specifications

HMI

Specifications	NB3Q		NB5Q		NB7W		NB10W
	TW00B	TW01B	TW00B	TW01B	TW00B	TW01B	TW01B
Display type	3.5" TFT LCD		5.6" TFT LCD		7" TFT LCD		10.1" TFT LCD
Display resolution (H × V)	320 × 240		320 × 234		800 × 480		800 × 480
Number of colours	65,536						
Backlight	LED						
Backlight lifetime	50,000 hours of operating time at the normal temperature (25°C)*1						
Touch panel	Analogue resistive membrane, resolution 1024 × 1024, life: 1 million touch operations						
Dimensions in mm (H × W × D)	103.8 × 129.8 × 52.8		142 × 184 × 46		148 × 202 × 46		210.8 × 268.8 × 54.0
Weight	310 g max.	315 g max.	620 g max.	625 g max.	710 g max.	715 g max.	1,545 g max.

*1 This is the estimated time when the luminous intensity is decreased by 50% per LED at room temperature and humidity. It is a typical value.

Functionality

Specifications	NB3Q		NB5Q		NB7W		NB10W
	TW00B	TW01B	TW00B	TW01B	TW00B	TW01B	TW01B
Internal memory	128MB (including system area)						
Memory interface	–	USB Memory	–	USB Memory	–	USB Memory	USB Memory
Serial (COM1)	RS-232C/422A/485 (not isolated), Transmission distance: 15m Max. (RS-232C), 500m Max. (RS-422A/485), Connector: D-Sub 9-pin		RS-232C, Transmission distance: 15 m Max., Connector: D-Sub 9-pin				
Serial (COM2)	–		RS-232C/422A/485 (not isolated), Transmission distance: 15m Max. (RS-232C), 500m Max. (RS-422A/485), Connector: D-Sub 9-pin				
USB Host	Equivalent to USB 2.0 full speed, type A, Output power 5V, 150mA						
USB Slave	Equivalent to USB 2.0 full speed, type B, Transmission distance: 5m						
Printer connection	PictBridge support						
Ethernet	–	10/100 base-T	–	10/100 base-T	–	10/100 base-T	10/100 base-T

General

Specifications	NB3Q		NB5Q		NB7W		NB10W
	TW00B	TW01B	TW00B	TW01B	TW00B	TW01B	TW01B
Line voltage	20.4 to 27.6 VDC (24 VDC –15 to 15%)						
Power consumption	5 W	9 W	6 W	10 W	7 W	11 W	14 W
Battery lifetime	5 years (at 25°C)						
Enclosure rating (front side)	Front operation part: IP65 (Dust proof and drip proof only from the front of the panel)						
Obtained standards	EC Directives, KC, cUL508						
Operating environment	No corrosive gases.						
Noise immunity	Compliant with IEC61000-4-4, 2KV (Power cable)						
Ambient operating temperature	0 to 50°C						
Ambient operating humidity	10% to 90% RH (without condensation)						

Applicable Controllers

Brand	Series
OMRON	Omron C Series Host Link
	Omron CJ/CS Series Host Link
	Omron CP Series
Mitsubishi	Mitsubishi Q_QnA (Link Port)
	Mitsubishi FX-485ADP/485BD/422BD (Multi-station)
	Mitsubishi FX0N/1N/2N/3G
	Mitsubishi FX1S
	Mitsubishi FX2N-10GM/20GM
	Mitsubishi FX3U
	Mitsubishi Q series (CPU Port)
	Mitsubishi Q00J (CPU Port)
	Mitsubishi Q06H
Panasonic	FP series
Siemens	Siemens S7-200
	Siemens S7-300/400 (PC Adapter Direct)
Allen-Bradley*1 (Rockwell)	AB DF1 AB CompactLogix/ControlLogix

Brand	Series
Schneider	Schneider Modicon Uni-TelWay
	Schneider Twido Modbus RTU
Delta	Delta DVP
LG (LS)	LS Master-K Cnet
	LS Master-K CPU Direct
	LS Master-K Modbus RTU
	LS XGT CPU Direct
	LS XGT Cnet
GE Fanuc Automation*1	GE Fanuc Series SNP GE SNP-X
Modbus	Modbus ASCII
	Modbus RTU
	Modbus RTU Slave
	Modbus RTU Extend
	Modbus TCP

*1 AB and GE will be supported by NB-Designer version 1.20 or higher.

Note: For details, refer to NB Series Host Connection Manual (Cat.No V108).

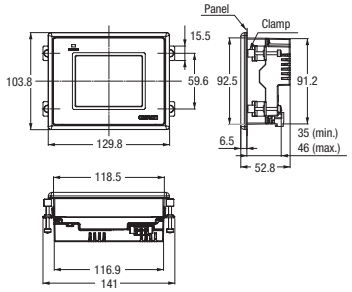
OMRON

NB series

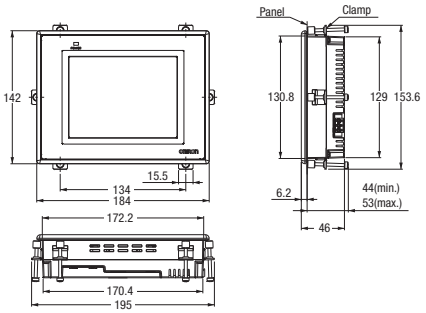
Dimensions

(Units: mm)

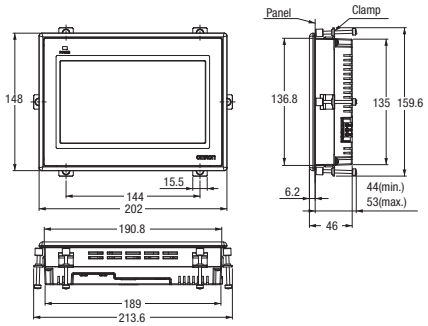
NB3Q



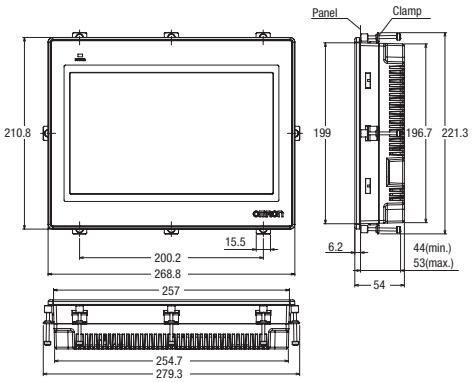
NB5Q



NB7W



NB10W



Model	Panel cutout (H × V mm)
NB3Q	119.0 (+0.5/–0) × 93.0 (+0.5/–0)
NB5Q	172.4 (+0.5/–0) × 131.0 (+0.5/–0)
NB7W	191.0 (+0.5/–0) × 137.0 (+0.5/–0)
NB10W	258.0 (+0.5/–0) × 200.0 (+0.5/–0)

Applicable panel thickness: 1.6 to 4.8 mm

Related Manuals

Cat. No	Model	Name
V106	NB-Designer	NB Series NB-Designer Operation Manual
V107	NB3Q, NB5Q, NB7W, NB10W	NB Series Setup Manual
V108	NB3Q, NB5Q, NB7W, NB10W	NB Series Host Connection Manual
V109	NB3Q, NB5Q, NB7W, NB10W	NB Series Startup Guide

NB series

Ordering information

Programmable Terminals

Product name	Specifications	Order code
NB3Q	3.5 inch, TFT LCD, Colour, 320 × 240 dots	NB3Q-TW00B
	3.5 inch, TFT LCD, Colour, 320 × 240 dots, USB Host, Ethernet	NB3Q-TW01B
NB5Q	5.6 inch, TFT LCD, Colour, 320 × 234 dots	NB5Q-TW00B
	5.6 inch, TFT LCD, Colour, 320 × 234 dots, USB Host, Ethernet	NB5Q-TW01B
NB7W	7 inch, TFT LCD, Colour, 800 × 480 dots	NB7W-TW00B
	7 inch, TFT LCD, Colour, 800 × 480 dots, USB Host, Ethernet	NB7W-TW01B
NB10W	10.1 inch, TFT LCD, Colour, 800 × 480 dots, USB Host, Ethernet	NB10W-TW01B

Options

Product name	Specifications	Order code
NB-to-PLC Connecting cable	For NB to PLC via RS-232C (CP/CJ/CS), 2m	XW2Z-200T
	For NB to PLC via RS-232C (CP/CJ/CS), 5m	XW2Z-500T
	For NB to PLC via RS-422A/485, 2m	NB-RSEXT-2M
Software	Supported Operating Systems: Windows 7, Windows Vista®, Windows XP[*1] (SP1 or higher). Download from Omron's regional websites.	NB-Designer[*2]
Display protective sheets	For the NB3Q contains 5 sheets	NB3Q-KBA04
	For the NB5Q contains 5 sheets	NB5Q-KBA04
	For the NB7W contains 5 sheets	NB7W-KBA04
	For the NB10W contains 5 sheets	NB10W-KBA04
Attachment	Mounting bracket for NT31/NT31C series to NB5Q series	NB5Q-ATT01

[*1] Except for Windows XP 64-bit version
[*2] The NB5Q-TW01B and NB7W-TW01B are supported by NB-Designer version 1.10 or higher.
The NB3Q-TW0_B and NB10W-TW01B are supported by NB-Designer version 1.20 or higher.

Apêndice A

Linguagem Ladder

A.0.1 Instruções do tipo Relé

Este tipo de instrução permite observar o estado *(ON/OFF)* de uma entrada, de uma saída ou de um ponto interno de entrada ou saída. [27]

Início de Ramificação

O início de uma ramificação é a primeira instrução que se coloca quando se pretende realizar uma equação lógica OR. Assim, é necessário, dentro de cada degrau, iniciar cada um dos ramos por ele constituintes. O início de ramificação está retratado na Figura A.1.

Figura A.1: Início de Ramificação

Fim de Ramificação

O fim de uma ramificação, como o nome indica, será colocado no fim do caminho da continuidade lógica de um degrau. Em cada degrau é necessário terminar todo os ramos que constituem essa equação lógica. O fim de ramificação está representado na Figura A.2.

Figura A.2: Fim de Ramificação

Contacto normalmente aberto

Um contacto contém um endereço de memória associado que pode ser usado como saída, como entrada ou como um ponto interno. O contacto normalmente aberto, como o nome indica, quando o estado está aberto *(ON)*, o contacto está fechado e existe continuidade lógica. Se o estado estiver

desligado *(OFF)*, ocorre o oposto, o contacto abre e para a continuidade lógica. O contacto normalmente aberto está retratado na Figura A.3.

Figura A.3: Contacto normalmente aberto

Contacto normalmente fechado

O contacto normalmente fechado, como o nome indica, quando o estado está desligado *(OFF)*, o contacto está fechado e existe continuidade lógica. Se o estado estiver aberto *(ON)* existe o oposto, o contacto abre e para a continuidade lógica. O contacto normalmente fechado, pode observar-se na Figura A.4.

Figura A.4: Contacto normalmente fechado

Saída normalmente aberta

Uma saída tem um endereço de memória que pode representar uma saída ou um ponto interno. Se a saída se encontrar ativada, isto é, se existir continuidade lógica, será ativada essa saída ou ponto interno. Caso não exista continuidade lógica, a saída ou ponto interno estarão desligados. Por exemplo, se esta saída estiver endereçada para um motor, o mesmo será ligado quando a saída estiver ativa. A saída normalmente aberta pode observar-se na Figura A.5.

Figura A.5: Saída normalmente aberta

Saída normalmente fechada

O princípio de funcionamento é parecido ao anterior, tendo uma lógica inversa. Se a saída se encontrar ativada, isto é, se existir continuidade lógica, será desligada essa saída ou ponto interno. Caso não exista continuidade lógica, a saída ou ponto interno estarão ligados. A saída normalmente fechada está representada na Figura A.6.

Figura A.6: Saída normalmente fechada

A.0.2 Instruções de temporização e contagem

Em muitas tarefas de controlo é necessário controlar o tempo. Isto é, é necessário ligar ou desligar um dispositivo ao fim de determinado tempo ou contagem. Para tal, os PLCs têm embutidas instruções de temporizadores e contagem, que têm funções semelhantes às dos temporizadores e contadores mecânicos ou eletrónicos. O temporizador conta os instantes de tempo necessários para a duração pretendida, enquanto que o contador regista o número de ocorrências. As duas instruções, contagem e temporização, utilizam dois registos, nomeadamente, um para armazenar o número de contagens e outro para armazenar o valor inicial. [26]

Para explicar o funcionamento dos temporizadores, está apresentado um pequeno exemplo de um temporizador na Figura A.7, estando configurado para um tempo de 60 segundos. Assim, um temporizador é iniciado quando é pressionado o botão "X0". O primeiro registo armazena o número de contagens (tempo) já efetuadas e o segundo registo armazena o valor desejado para esperar. Quando "Y0"desliga, o temporizador é ativado e, por isso, é necessário esperar que as contagens atinjam o valor desejado. Quando isso ocorre, o programa continua. [31]

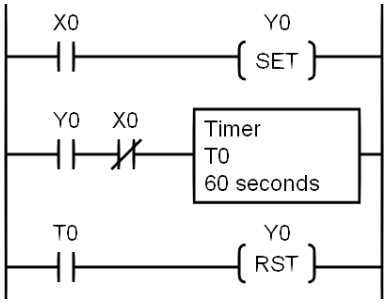


Figura A.7: Lógica ladder de um Timer [31]

Para explicar o funcionamento dos contadores é apresentado um pequeno exemplo de um contador na Figura A.8. O mesmo, está configurado para um valor de *reset* em 10. Quando a entrada "X0" é ativada, o valor de CT0 do contador é incrementado uma unidade. Sempre que o valor de CT0 atinge o valor de *reset*, é ativado o bit de status CT0, e acionada a saída Y0. O bit de status de CT0 permanece ativo até que o contador reinicie, isto é, seja ativada a entrada "X1". [32]

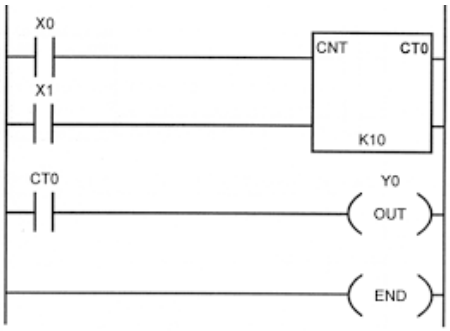


Figura A.8: Lógica ladder de um contador [32]

Existem diferentes tipos de *timmers*:

- **On-delay (TON)** - No diagrama temporal da Figura A.9a , podem analisar-se os sinais de condição de entrada e o sinal de saída. Assim, quando a entrada é ativada (0 para 1), este componente espera X segundos antes de ativar a saída.
- **Off-delay (TOF)** - No diagrama temporal da Figura A.9b, podem analisar-se os sinais de condição de entrada e o sinal de saída. Assim, quando a entrada é desligada (1 para 0), o temporizador inicia a contagem do tempo ficando x segundos ligado antes de desligar.
- **Pulse (TP)** - No diagrama temporal da Figura A.9c, podem analisar-se os sinais de condição de entrada e o sinal de saída. Assim, quando a entrada é ligada (0 para 1), o temporizador coloca o seu valor de saída a 1 por um período de tempo definido. Quando o tempo passa o valor de saída, volta para 0.

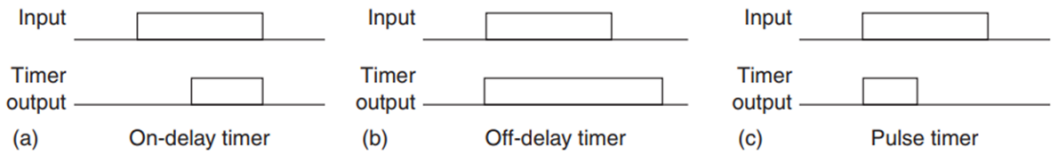


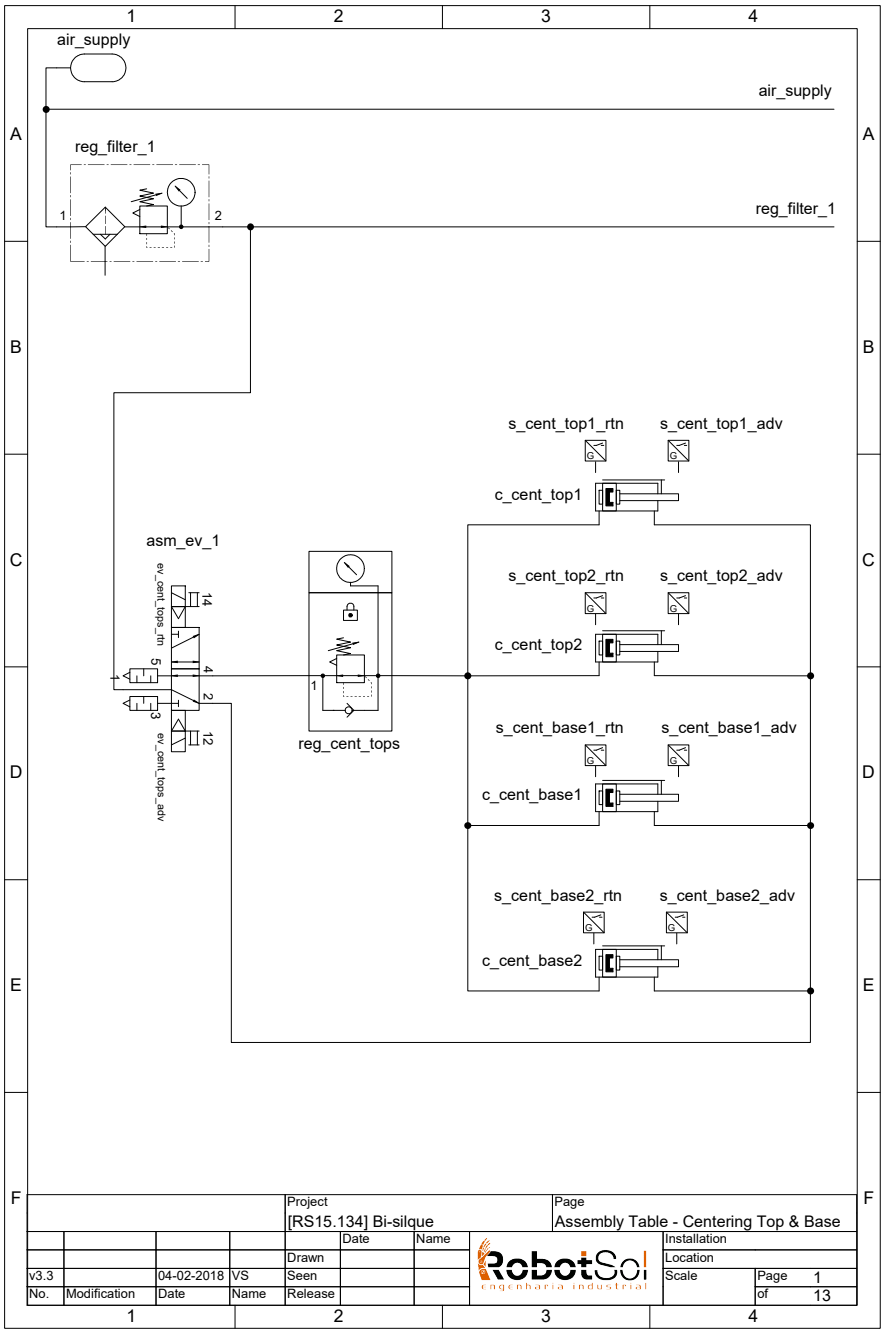
Figura A.9: Timmers ou Temporizadores [33]

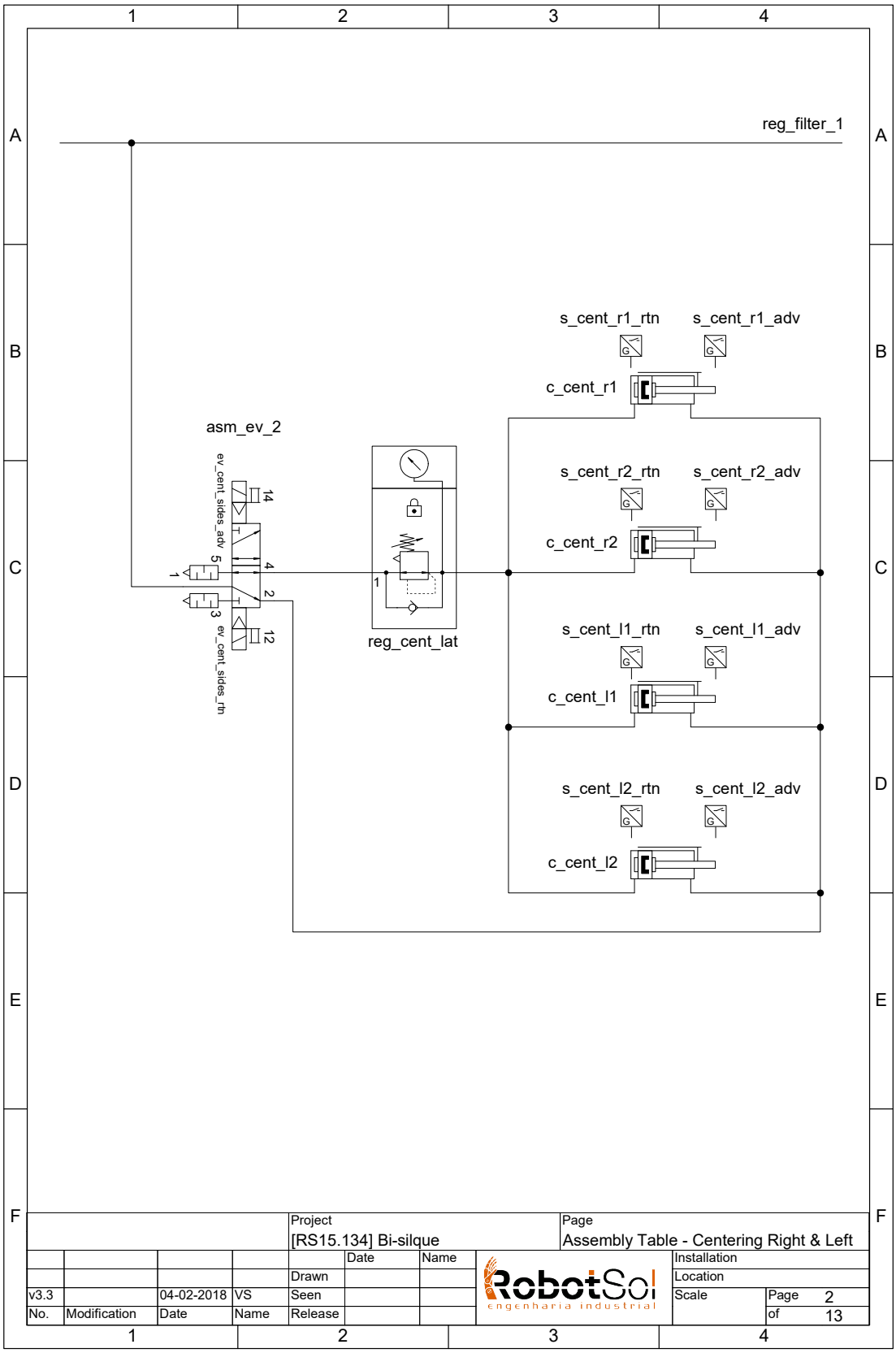
Ao contrário dos temporizadores, os contadores continuam a contar os eventos mesmo após o seu valor de contagem ser atingido. Visto isto, existem diferentes tipos de contadores, são eles:

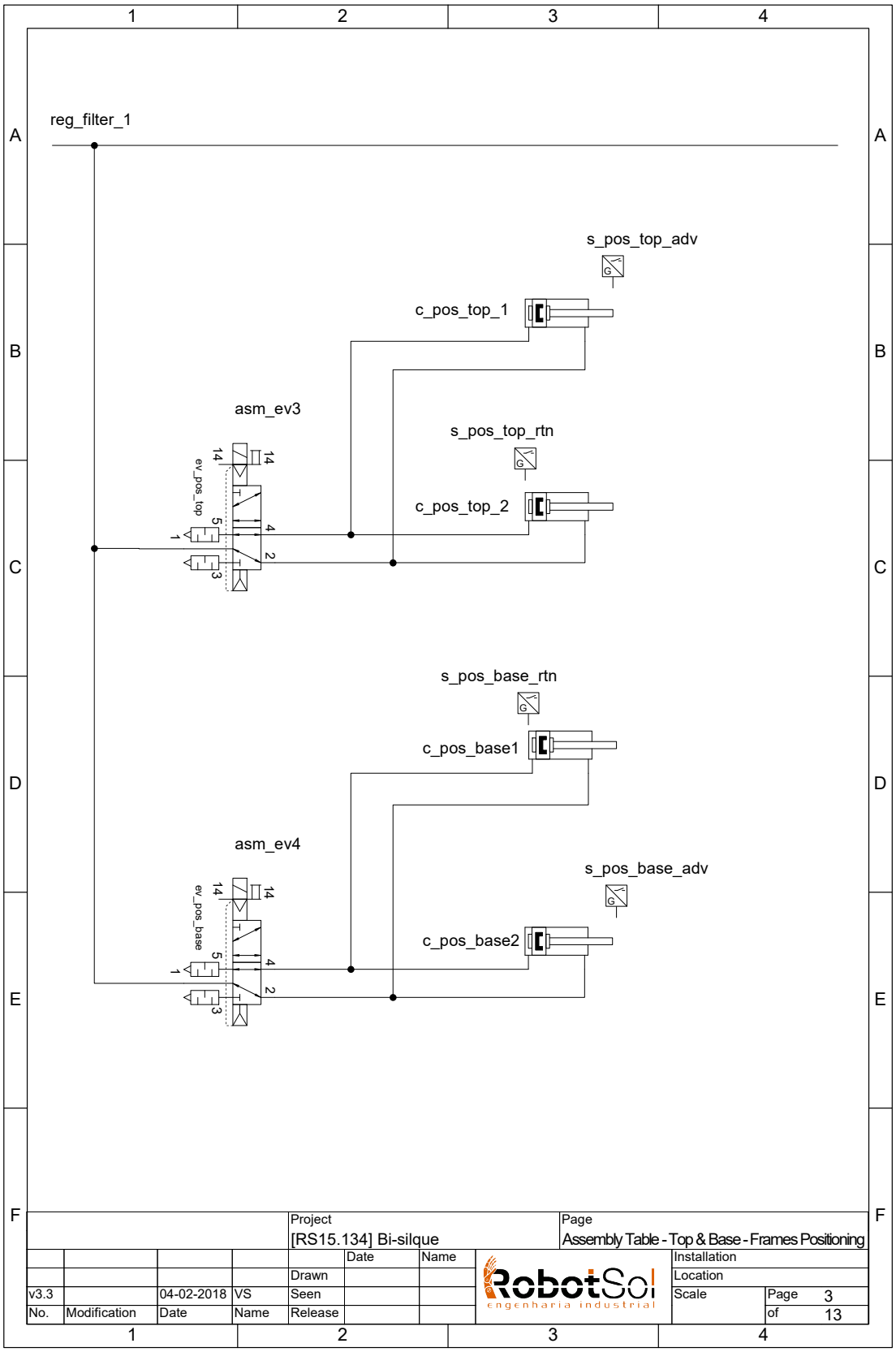
- **Down-Counters (CTD)** - O contador CTD decrementa uma unidade ao seu valor pré-programado. Sempre que o contador é ativado, isto é, ocorre um evento de contagem, o valor decresce até chegar ao zero.
- **Up-Counters (CTU)** - O contador CTU incrementa uma unidade ao seu valor acumulado sempre que o contador é ativado, isto é, ocorre um evento de contagem.

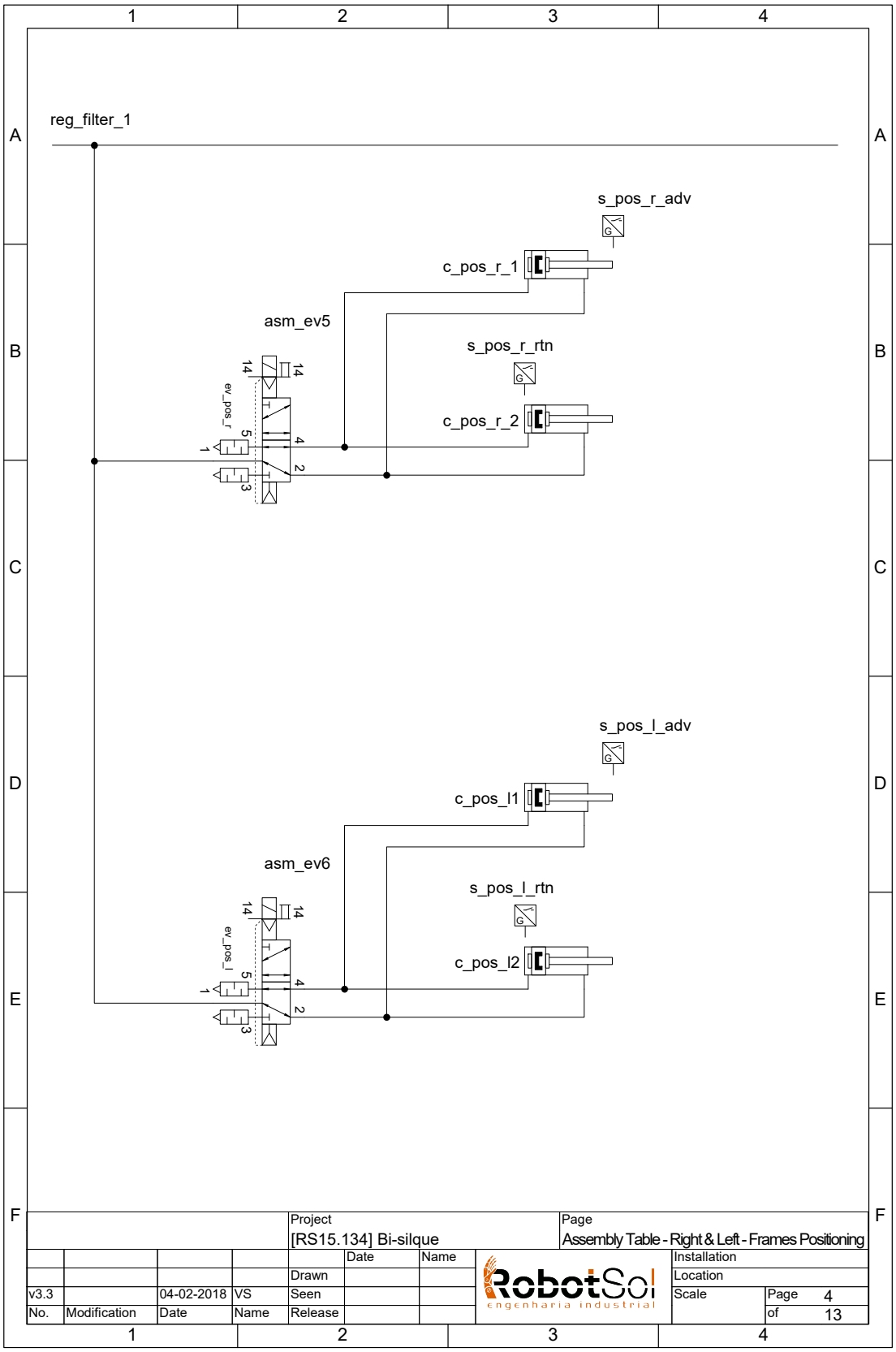
Apêndice B

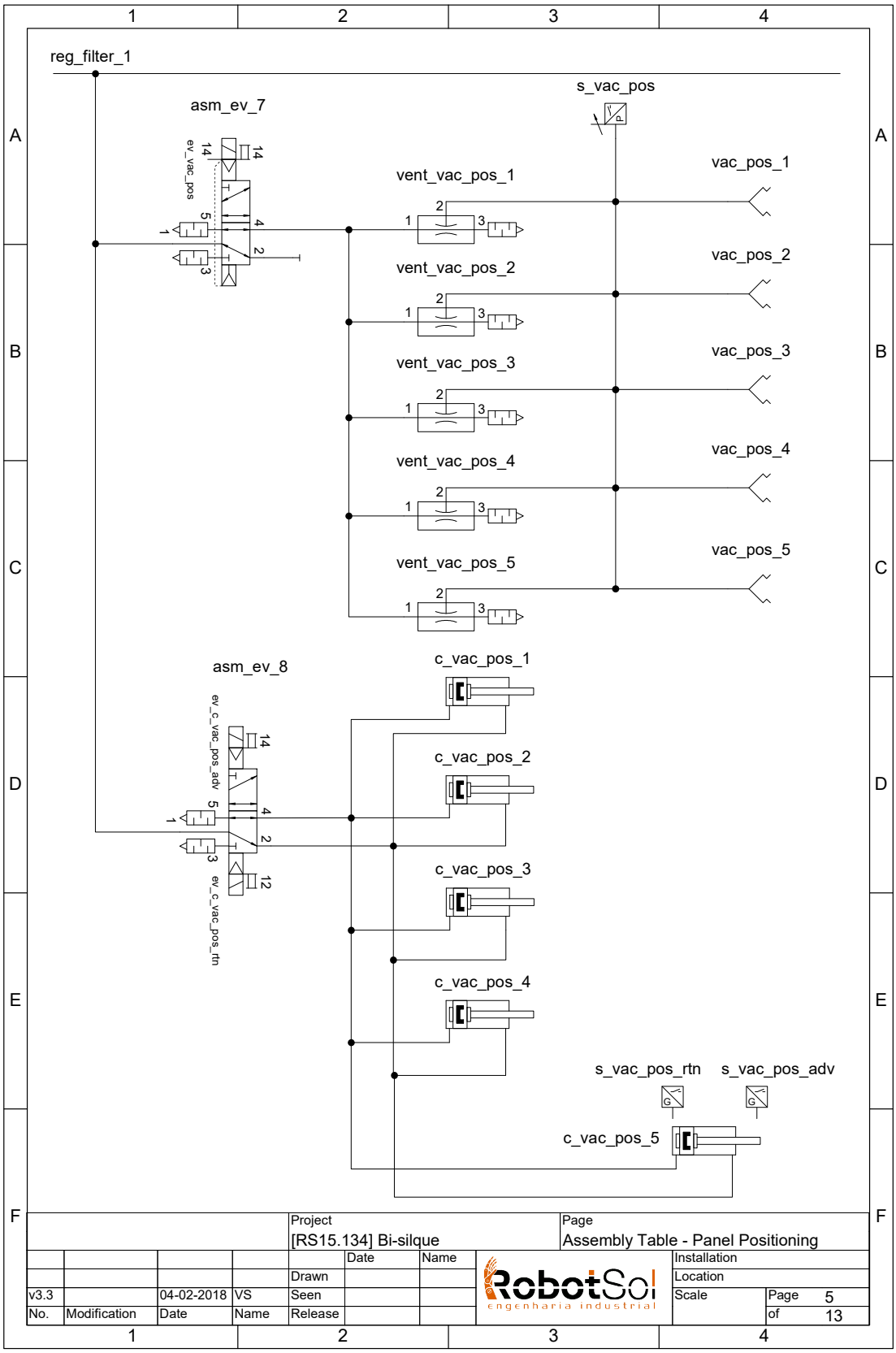
Circuito Pneumático

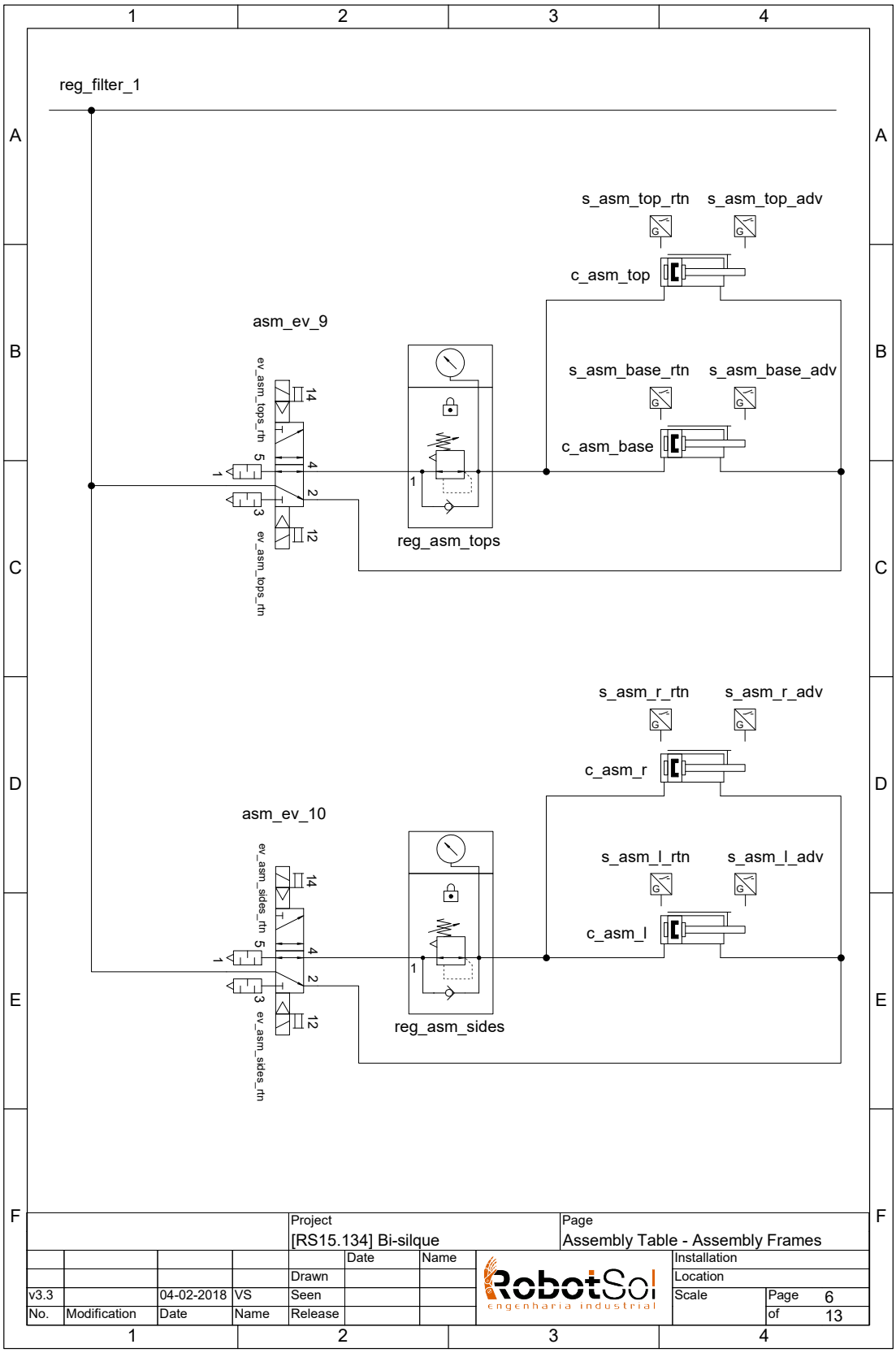


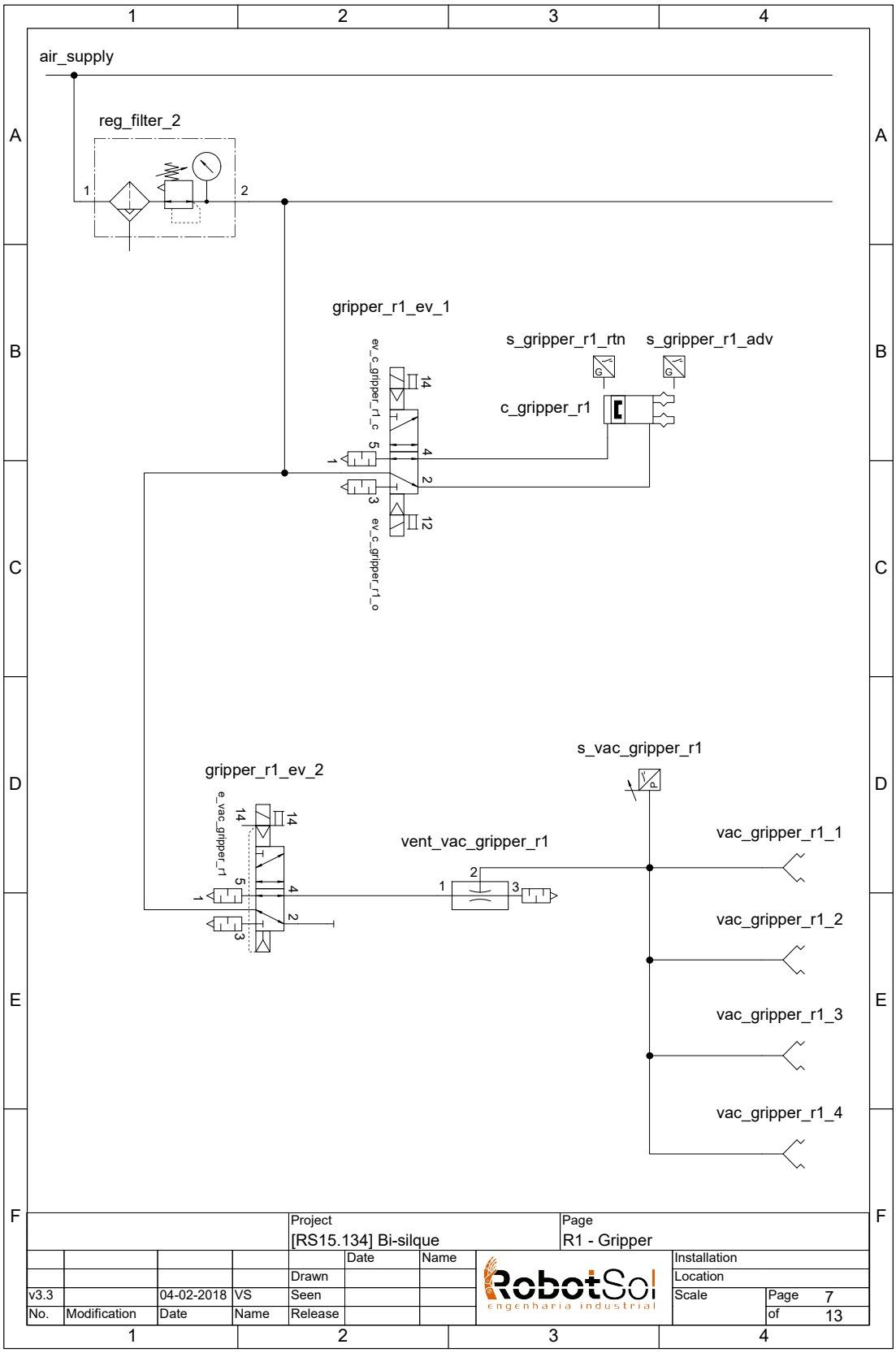

reg_filter_1
s_cent_r1_rtn
s_cent_r1_adv
c_cent_r1
asm_ev_2
ev_cent_sides_adv
14
5
4
1
2
3
12
ev_cent_sides_rtn
reg_cent_lat
s_cent_r2_rtn
s_cent_r2_adv
c_cent_r2
s_cent_l1_rtn
s_cent_l1_adv
c_cent_l1
s_cent_l2_rtn
s_cent_l2_adv
c_cent_l2
Project
[RS15.134] Bi-silque
Page
Assembly Table - Centering Right & Left
Date
Name
Drawn
Seen
Release
v3.3
04-02-2018
VS
No.
Modification
Date
Name
RobotSol
engenharia industrial
Installation
Location
Scale
Page 2
of 13

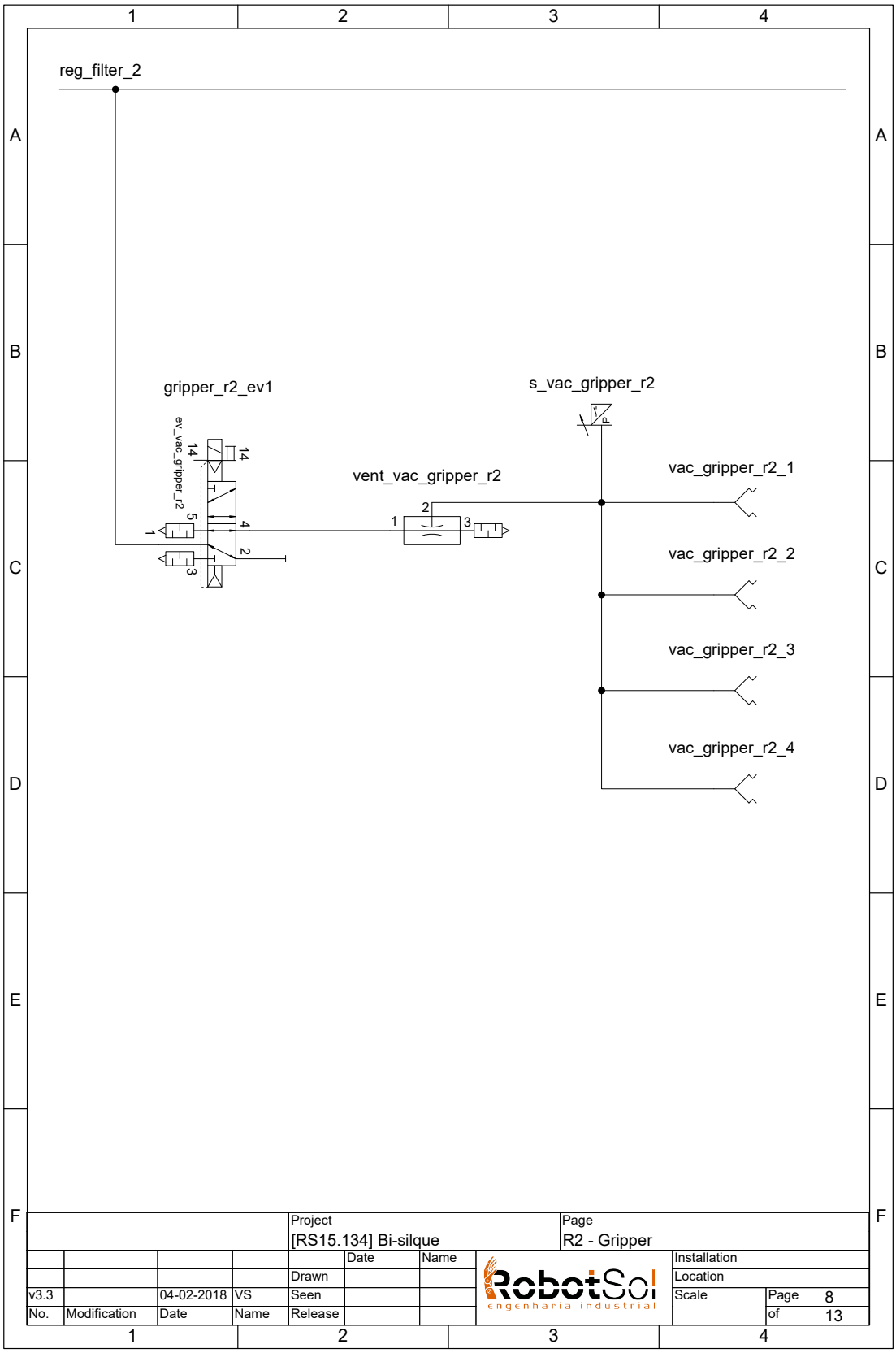

reg_filter_1
s_pos_top_adv
c_pos_top_1
asm_ev3
s_pos_top_rtn
c_pos_top_2
ev_pos_top
s_pos_base_rtn
c_pos_base1
asm_ev4
s_pos_base_adv
c_pos_base2
ev_pos_base
Project
[RS15.134] Bi-silque
Page
Assembly Table - Top & Base - Frames Positioning
Date
Name
Drawn
Seen
Release
RobotSol
engenharia industrial
Installation
Location
Scale
Page 3
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name

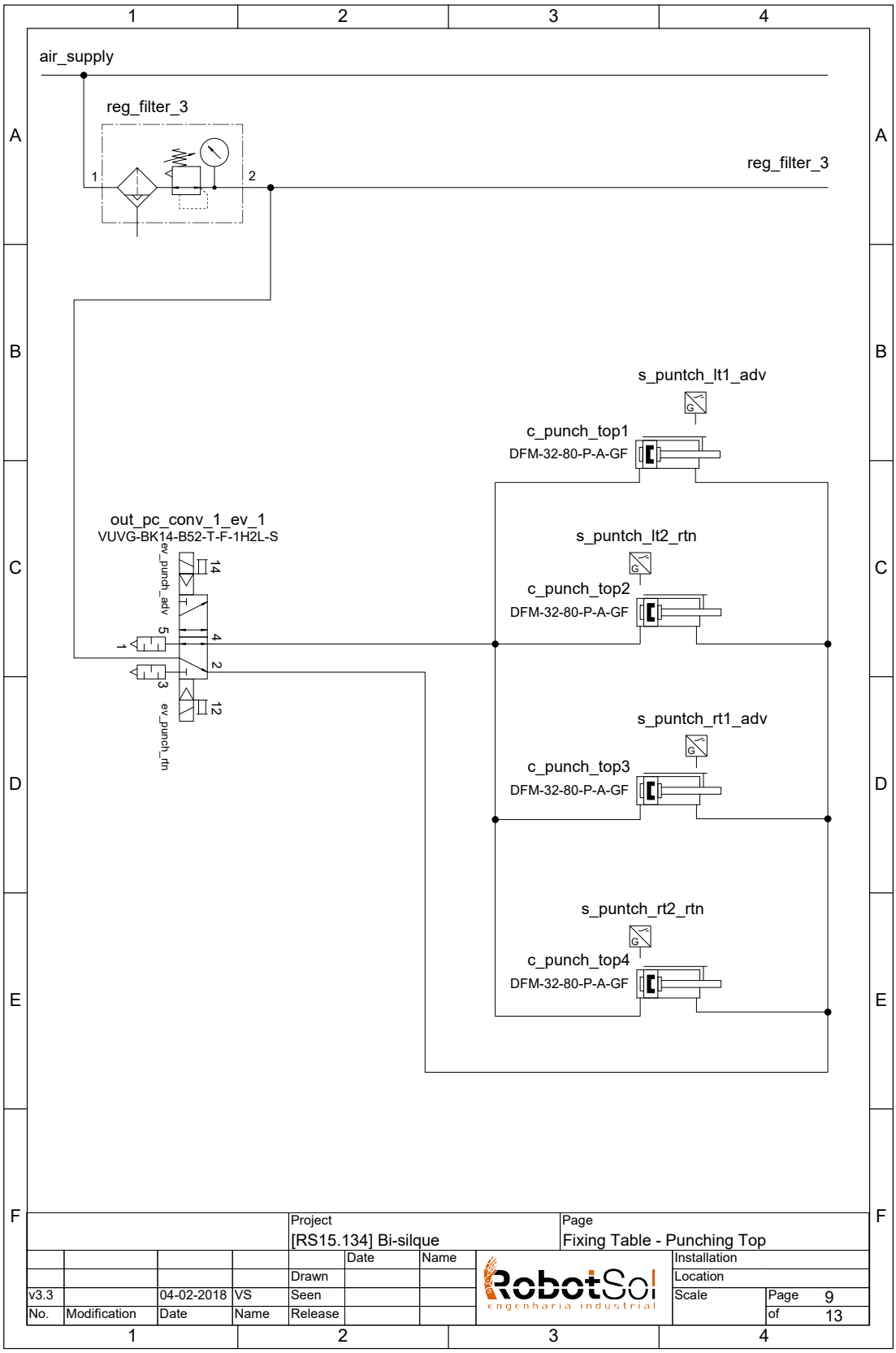

reg_filter_1
s_pos_r_adv
c_pos_r_1
asm_ev5
s_pos_r_rtn
c_pos_r_2
ev_pos_r
s_pos_l_adv
c_pos_l1
asm_ev6
s_pos_l_rtn
c_pos_l2
ev_pos_l
Project
[RS15.134] Bi-silque
Page
Assembly Table - Right & Left - Frames Positioning
Date
Name
Drawn
Seen
Release
RobotSol
engenharia industrial
Installation
Location
Scale
Page 4
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name

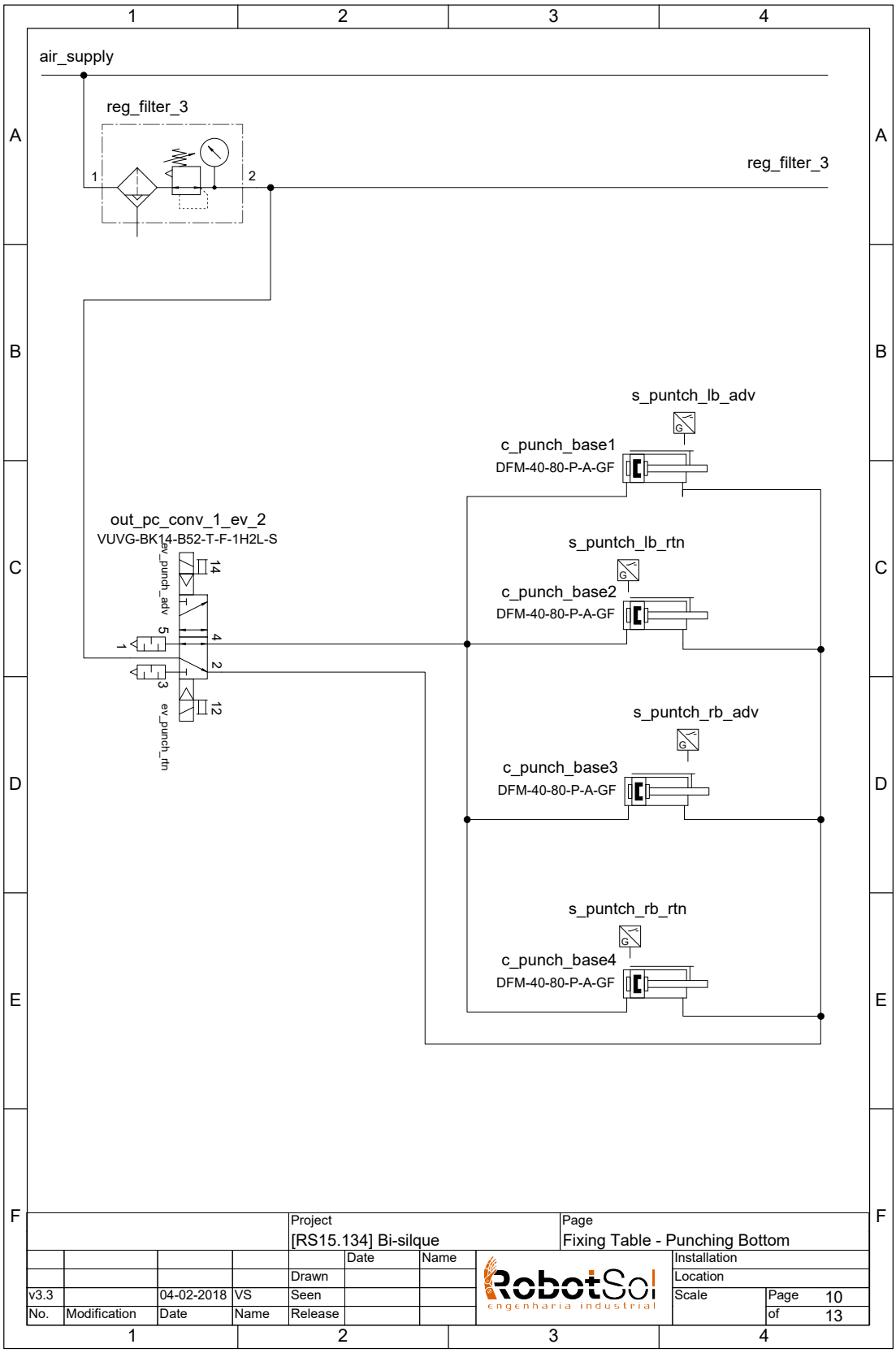

reg_filter_1
asm_ev_7
ev_vac_pos
s_vac_pos
vent_vac_pos_1
vent_vac_pos_2
vent_vac_pos_3
vent_vac_pos_4
vent_vac_pos_5
vac_pos_1
vac_pos_2
vac_pos_3
vac_pos_4
vac_pos_5
asm_ev_8
ev_c_vac_pos_adv
ev_c_vac_pos_rtn
c_vac_pos_1
c_vac_pos_2
c_vac_pos_3
c_vac_pos_4
c_vac_pos_5
s_vac_pos_rtn
s_vac_pos_adv
Project
[RS15.134] Bi-silque
Page
Assembly Table - Panel Positioning
Date
Name
Drawn
Seen
Release
RobotSol
engenharia industrial
Installation
Location
Scale
Page 5
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name

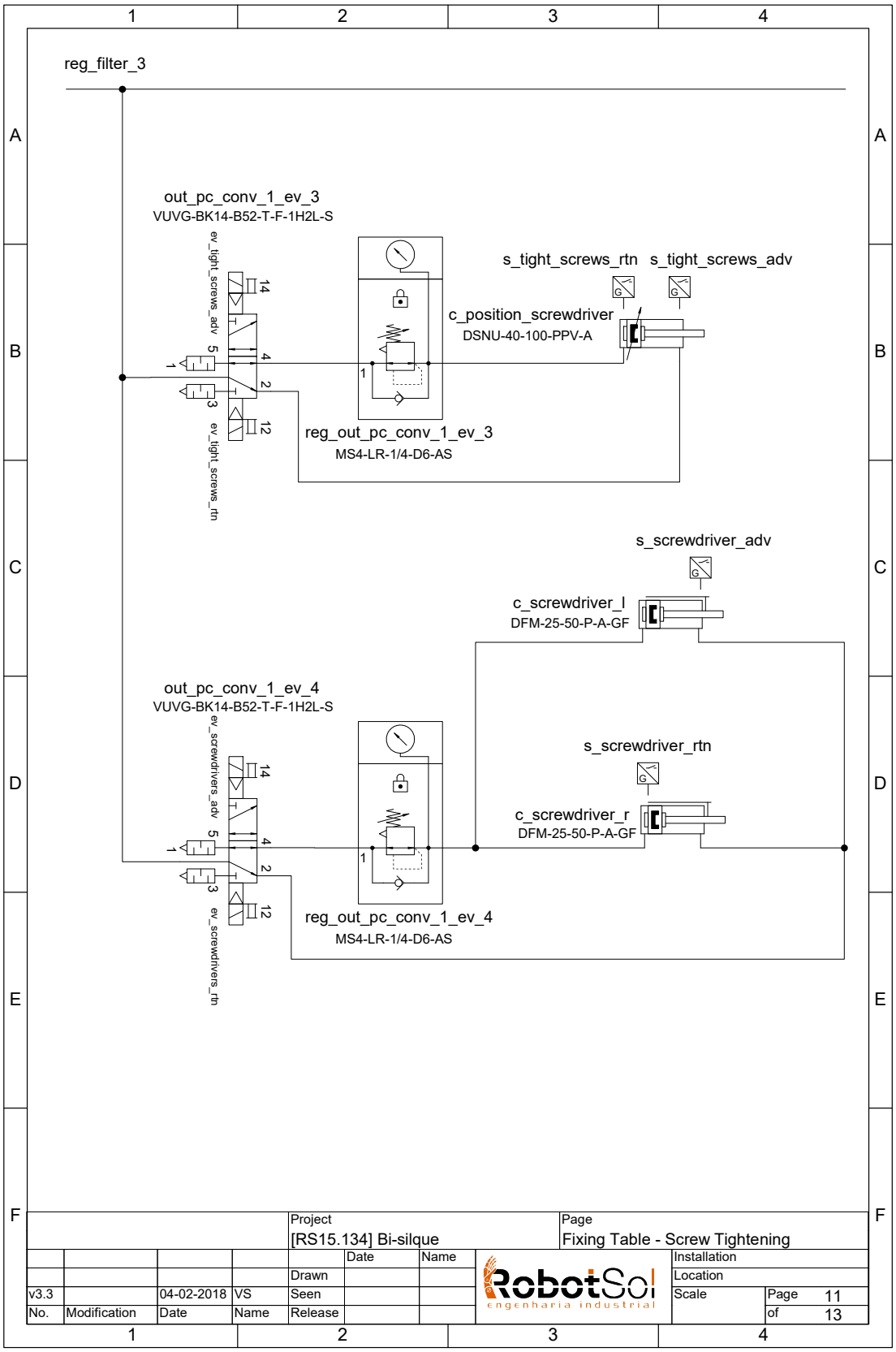

reg_filter_1
s_asm_top_rtn
s_asm_top_adv
c_asm_top
asm_ev_9
s_asm_base_rtn
s_asm_base_adv
c_asm_base
ev_asm_tops_rtn
reg_asm_tops
ev_asm_tops_rtn
s_asm_r_rtn
s_asm_r_adv
c_asm_r
asm_ev_10
s_asm_l_rtn
s_asm_l_adv
c_asm_l
ev_asm_sides_rtn
reg_asm_sides
ev_asm_sides_rtn
Project
[RS15.134] Bi-silque
Page
Assembly Table - Assembly Frames
Date
Name
Drawn
Seen
Release
v3.3
04-02-2018
VS
No.
Modification
Date
Name
RobotSol
engenharia industrial
Installation
Location
Scale
Page 6
of 13

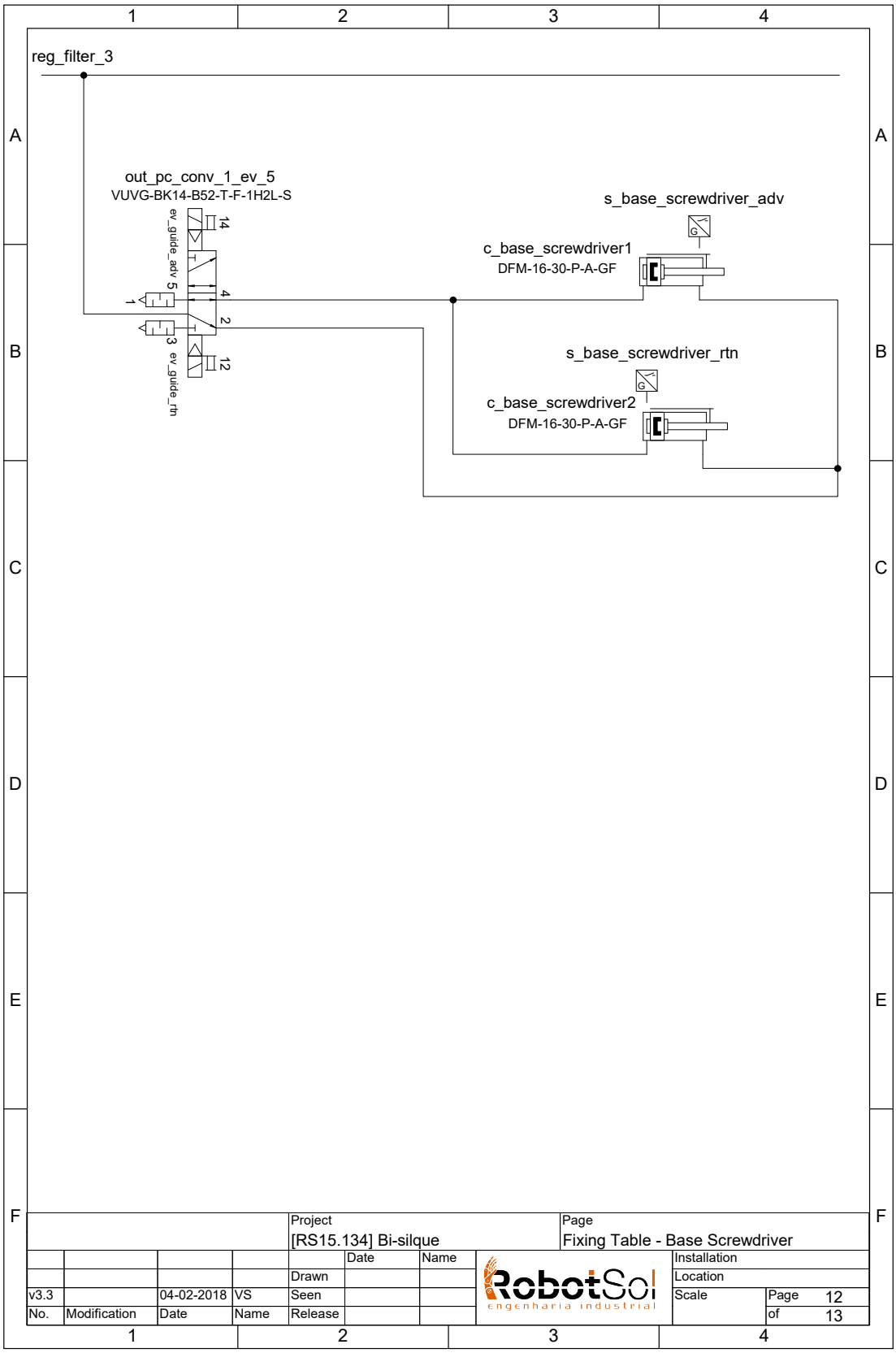

air_supply
reg_filter_2
gripper_r1_ev_1
s_gripper_r1_rtn
s_gripper_r1_adv
c_gripper_r1
ev_c_gripper_r1_c
ev_c_gripper_r1_o
gripper_r1_ev_2
e_vac_gripper_r1
s_vac_gripper_r1
vent_vac_gripper_r1
vac_gripper_r1_1
vac_gripper_r1_2
vac_gripper_r1_3
vac_gripper_r1_4
Project
[RS15.134] Bi-silque
Page
R1 - Gripper
Drawn
Seen
Release
Date
Name
v3.3
04-02-2018
VS
No.
Modification
Installation
Location
Scale
Page 7
of 13
RobotSol
engenharia industrial

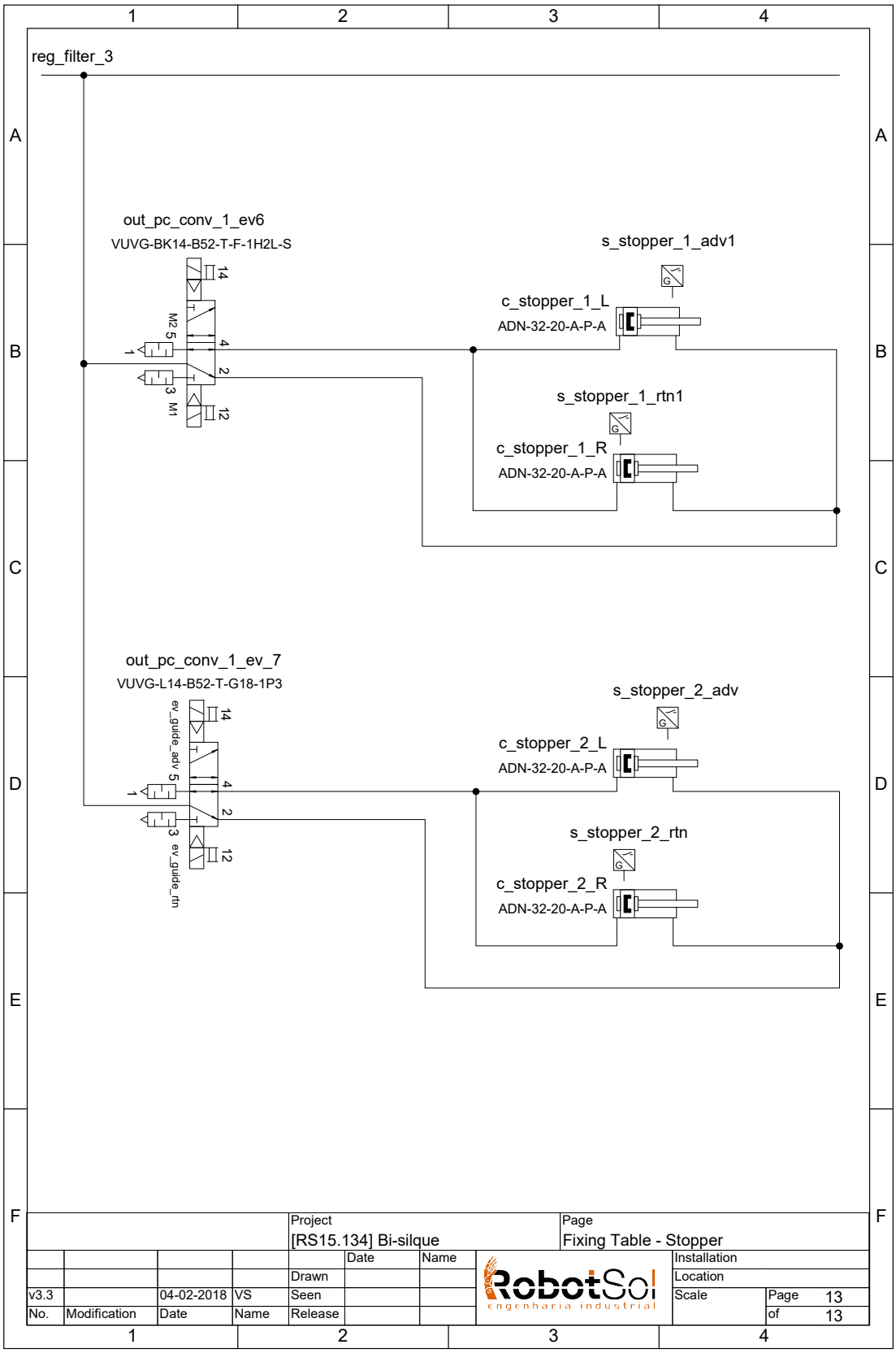

reg_filter_2
gripper_r2_ev1
ev_vac_gripper_r2
14
14
5
4
1
2
3
vent_vac_gripper_r2
2
1
3
s_vac_gripper_r2
vac_gripper_r2_1
vac_gripper_r2_2
vac_gripper_r2_3
vac_gripper_r2_4
Project
[RS15.134] Bi-silque
Page
R2 - Gripper
Date
Name
Drawn
Seen
Release
RobotSol
engenharia industrial
Installation
Location
Scale
Page 8
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name


air_supply
reg_filter_3
reg_filter_3
1
2
s_puntch_lt1_adv
c_punch_top1
DFM-32-80-P-A-GF
out_pc_conv_1_ev_1
VUVG-BK14-B52-T-F-1H2L-S
ev_puntch_adv
ev_puntch_rtn
14
12
5
1
4
2
3
s_puntch_lt2_rtn
c_punch_top2
DFM-32-80-P-A-GF
s_puntch_rt1_adv
c_punch_top3
DFM-32-80-P-A-GF
s_puntch_rt2_rtn
c_punch_top4
DFM-32-80-P-A-GF
Project
[RS15.134] Bi-silque
Page
Fixing Table - Punching Top
Date
Name
Drawn
Seen
Release
RobotSol
engenharia industrial
Installation
Location
Scale
Page 9
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name


air_supply
reg_filter_3
reg_filter_3
1
2
s_puntch_lb_adv
c_punch_base1
DFM-40-80-P-A-GF
out_pc_conv_1_ev_2
VUVG-BK14-B52-T-F-1H2L-S
ev_punch_adv
s_puntch_lb_rtn
c_punch_base2
DFM-40-80-P-A-GF
s_puntch_rb_adv
c_punch_base3
DFM-40-80-P-A-GF
s_puntch_rb_rtn
c_punch_base4
DFM-40-80-P-A-GF
ev_punch_rtn
Project
[RS15.134] Bi-silque
Page
Fixing Table - Punching Bottom
Drawn
Seen
Release
Date
Name
Installation
Location
Scale
Page 10
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name
RobotSol
engenharia industrial


reg_filter_3
out_pc_conv_1_ev_3
VUVG-BK14-B52-T-F-1H2L-S
ev_tight_screws_adv
ev_tight_screws_rtn
s_tight_screws_rtn
s_tight_screws_adv
c_position_screwdriver
DSNU-40-100-PPV-A
reg_out_pc_conv_1_ev_3
MS4-LR-1/4-D6-AS
s_screwdriver_adv
c_screwdriver_l
DFM-25-50-P-A-GF
out_pc_conv_1_ev_4
VUVG-BK14-B52-T-F-1H2L-S
ev_screwdrivers_adv
ev_screwdrivers_rtn
s_screwdriver_rtn
c_screwdriver_r
DFM-25-50-P-A-GF
reg_out_pc_conv_1_ev_4
MS4-LR-1/4-D6-AS
Project
[RS15.134] Bi-silque
Page
Fixing Table - Screw Tightening
Drawn
Date
Name
Seen
Release
RobotSol
engenharia industrial
Installation
Location
Scale
Page 11
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name


1
2
3
4
A
B
C
D
E
F
reg_filter_3
out_pc_conv_1_ev_5
VUVG-BK14-B52-T-F-1H2L-S
ev_guide_adv
ev_guide_rtn
14
12
5
1
3
4
2
s_base_screwdriver_adv
c_base_screwdriver1
DFM-16-30-P-A-GF
s_base_screwdriver_rtn
c_base_screwdriver2
DFM-16-30-P-A-GF
Project
[RS15.134] Bi-silque
Page
Fixing Table - Base Screwdriver
Date
Name
Drawn
Seen
Release
RobotSol
engenharia industrial
Installation
Location
Scale
Page 12
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name


reg_filter_3
out_pc_conv_1_ev6
VUVG-BK14-B52-T-F-1H2L-S
M2
M1
s_stopper_1_adv1
c_stopper_1_L
ADN-32-20-A-P-A
s_stopper_1_rtn1
c_stopper_1_R
ADN-32-20-A-P-A
out_pc_conv_1_ev_7
VUVG-L14-B52-T-G18-1P3
ev_guide_adv
ev_guide_rtn
s_stopper_2_adv
c_stopper_2_L
ADN-32-20-A-P-A
s_stopper_2_rtn
c_stopper_2_R
ADN-32-20-A-P-A
Project
[RS15.134] Bi-silque
Page
Fixing Table - Stopper
Drawn
Seen
Release
Date
Name
RobotSol
engenharia industrial
Installation
Location
Scale
Page 13
of 13
v3.3
04-02-2018
VS
No.
Modification
Date
Name

Apêndice C

Lista de Material

Tabela C.1: Material utilizado no Armazém (Pilha de entrada de painéis)

Armazém (Pilha de entrada de painéis)		
Quantidade	**Referência**	**Descrição**
1	O1D300	Sensor óptico de nível

Tabela C.2: Material utilizado no transportador de entrada

Transportador de entrada		
Quantidade	**Referência**	**Descrição**
1	OPU200	Fotocélula em forqueta

Tabela C.3: Material utilizado na mesa de fixação

Mesa de fixação				
Quantidade	**Referência**	**Sensor**		**Descrição**
		A	**R**	
1	DNC-32-150-PPV-A	x	x	Cilindro normalizado de dupla acção (DNC). Diâmetro do atuador de 32mm. Curso de 150mm, PPV amortecimento pneumático, ajustável nas posições finais de curso A Para sensor de proximidade
1	DSNU-40-100-PPV-A	x	x	Cilindro redondo (DSNU). Diâmetro do atuador de 40mm. Curso de 100mm, PPV amortecimento pneumático, ajustável nas posições finais de curso A Para sensor de proximidade
2	DFM-25-50-P-A-GF		x	Atuador de dupla ação (DFM).Diâmetro do atuador de 25mm. Curso de 50mm. P Anéis/placas de amortecimento elástico nas posições finais. A Para sensor de proximidade.
		x	x	
2	DFM-16-25-P-A-GF	x	x	Atuador de dupla ação (DFM). Diâmetro do atuador de 16mm. Curso de 25mm. P Anéis/placas de amortecimento elástico nas posições finais. A Para sensor de proximidade.
		x	x	
4	DFM-32-80-P-A-GF		x	Atuador de dupla ação DFM. Diâmetro do atuador de 32mm. Curso de 80mm, P Anéis/placas de amortecimento elástico nas posições finais A Para sensor de proximidade GF Guia de buchas deslizantes
		x		
			x	
		x		
4	DFM-40-80-P-A-GF		x	Atuador de dupla ação DFM. Diâmetro do atuador de 40mm. Curso de 80mm, P Anéis/placas de amortecimento elástico nas posições finais A Para sensor de proximidade GF Guia de buchas deslizantes
		x		
			x	
		x		
2	MS4-LR-1/4-D6	-	-	Regulador de pressão com faixa de pressão de 0,3 a 7 bar (manual).
6	VUVG-B14-B52-T-F	-	-	Válvula: Bistável, Retorno: Sem
1	SMT-8M-A-PS-24V-E-0,3-M8D	-	-	Sensor de proximidade eletrônico
2	Automatic screw supplier - SF30A			Fornecedor de parafuso automático
2	Screwdriver - FP075			Aparafusadora pneumática industrial automática

Tabela C.4: Material utilizado na mesa de indexação

Mesa de indexação				
Quantidade	**Referência**	**Sensor**		**Descrição**
		A	**R**	
2	DSNU-32-100-PPV-A	x	x	Cilindro redondo DSNU, métrico. Diâmetro do atuador de 32mm.
		x	x	Curso de 100mm, PPV amortecimento pneumático, ajustável nas posições finais de curso
2	DSNU-32-50-PPV-A	x	x	Cilindro redondo DSNU, métrico. Diâmetro do atuador de 32mm.
		x	x	Curso de 50mm, PPV amortecimento pneumático, ajustável nas posições finais de curso
2	ADNGF-32-130-P-A	x	x	Cilindro compacto ADNGF, métrico. Diâmetro do atuador de 32mm.
		x	x	Curso de 130mm, P Anéis/placas de amortecimento elástico nas posições finais
2	ADNGF-32-180-P-A	x	x	Cilindro compacto ADNGF, métrico. Diâmetro do atuador de 32mm.
		x	x	Curso de 180mm, P Anéis/placas de amortecimento elástico nas posições finais
4	ADNGF-32-80-P-A	x	x	Cilindro compacto ADNGF, métrico. Diâmetro do atuador de 32mm. Curso de 80mm, P Anéis/placas de amortecimento elástico nas posições finais
		x	x	
		x	x	
		x	x	
8	CLR-12-10-R-P-A	x	-	Grampo linear / articulado CLR. Diâmetro do atuador de 12mm. Curso de 10mm. Movimento linear e giratório, 90 graus para a direita. Anéis / placas de amortecimento elástico nas posições finais
		-	x	
		-	x	
		x	-	
		x	-	
		-	x	
		x	-	
		-	x	
5	ADN-16-5-A-P-A	-	-	Cilindro compacto ADN. Diâmetro do atuador 16mm. Curso de 5mm. Anéis / placas de amortecimento elástico nas posições finais
		-	-	
		-	-	
		-	-	
		-	-	
1	SPAB-B2R-G18-2P-L1	-	-	Sensor de pressão SPAB, com mostrador de pressão
4	MS4-LR-1/4-D6	-	-	Regulador de pressão com faixa de pressão de 0,3 a 7 bar (manual)
8	VUVG-B14-B52-ZT-F-1P3	-	-	Válvula: Bistável, Retorno: Sem
	VUVG-B14-M52-AZT-F-1P3	-	-	Válvula: Monostável. Retorno: Mola pneumática
1	SPAB-P10R-G18-2P-L1	-	-	Sensor de pressão SPAB, com mostrador de pressão

Tabela C.5: Material utilizado no *gripper* do robô 1

Gripper R1		
Quantidade	**Referência**	**Descrição**
1	SPAE-V1R-Q4-PNLK-2.5K	Sensor de pressão com faixa de medição de pressão: 0 ... -1 bar
1	VUVG-L14-B52-T-G18-1P3	Eletroválvula biestável. Reset: Sem
1	VUVG-L14-M52-AT-G18-1P3	Eletroválvula monoestável. Reset: Mola pneumática
1	VN-30-L-T6-PQ4-VA5-RO2	Gerador de vácuo

Tabela C.6: Material utilizado no *gripper* do robô 2

Gripper R2		
Quantidade	**Referência**	**Descrição**
1	SPAE-V1R-Q4-PNLK-2.5K	Sensor de pressão com faixa de medição de pressão: 0 ... -1 bar
1	VUVG-L14-M52-AT-G18-1P3	Eletroválvula monoestável. Reset: Mola pneumática
1	VN-30-L-T6-PQ4-VA5-RO2	Gerador de vácuo

Apêndice D

Sequência Mesa Indexação

Inicialização

- Colocar tudo na posição inicial:

    Cilindros das ventosas recuados;

    Vácuo desligado;

    Centradores laterais recuados;

    Centradores topos recuados;

    Posicionadores laterais avançados;

    Posicionadores topos avançados;

    ASM laterais avançados;

    ASM topos avançados.

- Painel:

    Dar acesso ao robô p/ colocar o painel;

    Esperar a confirmação do robô em como este colocou o painel;

    Avançar centradores laterais;

    Avançar centradores topos;

    Ligar vácuo e subir ventosas;

    Recuar centradores topos;

    Recuar centradores laterais;

    Descer ventosas;

- Frames laterais:

    Dar acesso ao robô p/ colocar frames laterais;

    Esperar a confirmação do robô em como este colocou as frames;

    Recuar posicionador laterais;

    Recuar ASM laterais p/ fazer o encaixe das frames laterais;

- Frames topos:

    Dar acesso ao robô p/ colocar frames topos;

Esperar a confirmação do robô em como este colocou as frames;

Desligar vácuo;

Recuar posicionadores dos topos;

Recuar ASM topos p/ fazer o encaixe das frames topos;

- Finalização:

  Avançar posicionadores topos e laterais;

  Desligar vácuo;

  Avançar cilindros das ventosas;

  Avançar ASM laterais p/ libertar o quadro para o robô pegar;

  Avançar ASM tops p/ libertar o quadro para o robô pegar;

  Dar acesso ao robô 2;

  Esperar confirmação em como o robô 2 retirou o painel.

  Avançar posicionadores topos e laterais;

  Desligar vácuo;

  Avançar cilindros das ventosas;

  Avançar ASM tops p/ libertar o quadro para o robô pegar;

  Recuar ASM tops p/ libertar o quadro para o robô pegar;

  Avançar ASM tops p/ libertar o quadro para o robô pegar;

  Recuar ASM tops p/ libertar o quadro para o robô pegar;

Apêndice E

Sequência Inicial Da Mesa De Fixação

- Inicialização

  Colocar tudo na posição inicial:

  Vácuo desligado

  Cilindros mesa recuados (menos cilindro da barreira)

  Avançar Cilindro Barreira.

- Quadro:

  Dar acesso ao robô para ir apanhar o quadro completo

  Esperar a confirmação do robô em como este colocou o quadro na posição final.

- Aparafusar:

  Avançar cilindro topo aparafusadora

  Avançar cilindros base aparafusadoras

  Avançar cilindros aparafusadora

  Timer (esperar que aparafuse).

- Subir aparafusadora:

  Recuar cilindros aparafusadora

  Recuar cilindro topo aparafusadora

  Recuar cilindros base aparafusadoras

- Deslocar quadro:

  Motor_1 ON

  Sensor_1 desativado

  Sensor_2 activado

  Timmer (para o quadro bater no *stopper*)

  Motor_1 OFF.

- Amolgar quadro:

  Avançar cilindro *push* base

  Avançar cilindro *push* topo

Recuar cilindro *stopper*.

- Aparafusar:

  Avançar cilindros topo aparafusadora

  Avançar cilindros base aparafusadora

  Avançar cilindros aparafusadora

  Timer (esperar que aparafuse).

- Subir aparafusadora:

  Recuar cilindros topo aparafusadora

  Recuar cilindros aparafusadoras

  Recuar cilindros base aparafusadora.

- Descer *pusher*:

  Recuar *pusher* topo

  Recuar *pusher* base.

- Deslocar quadro:

  Motor_1 ON

  Sensor_2 continua activado

  Sensor_3 ON

  Motor_2 ON.

- Passadeira saída:

  Sensor_2 OFF

  Motor_1 OFF

  Sensor_3 OFF

  Motor_2 OFF

  Avançar *stopper*.

Apêndice F

Erros Na Primeira Solução

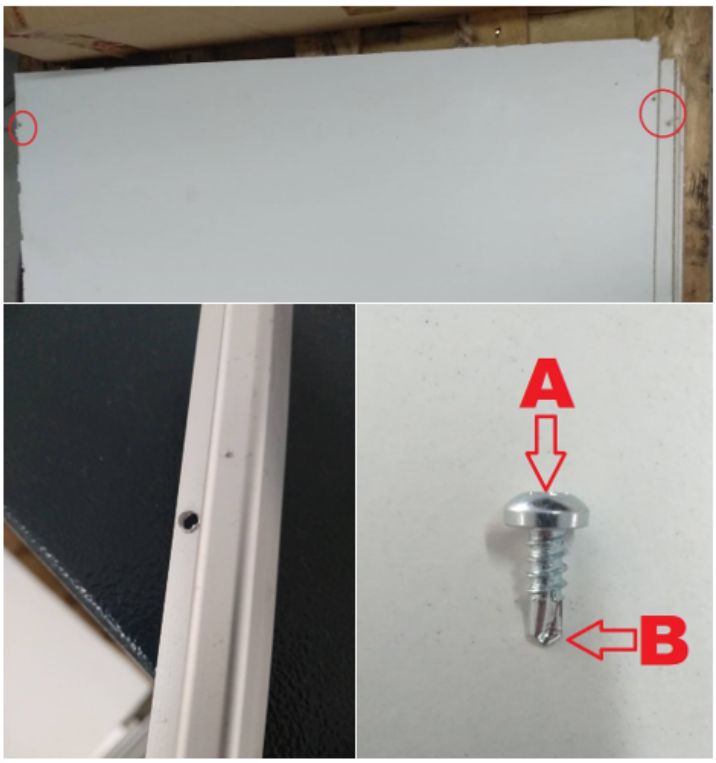
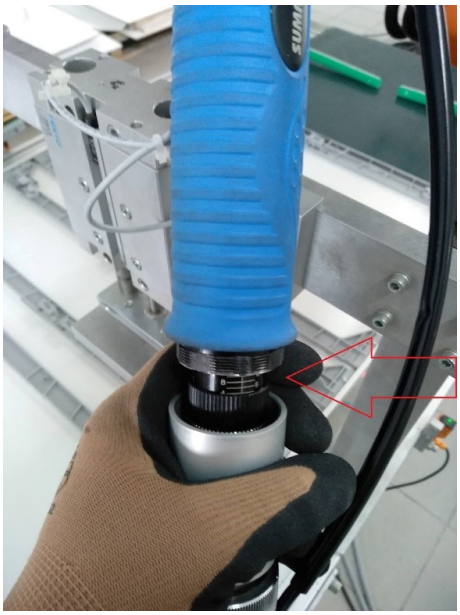


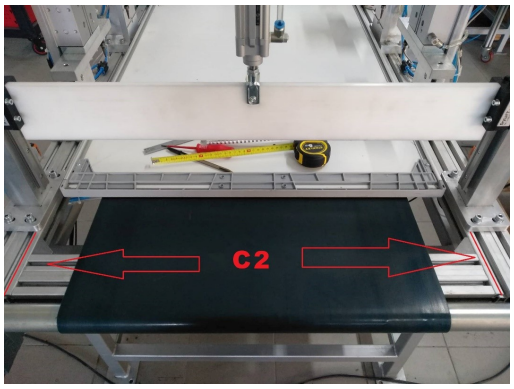
Figura F.1: Material desgastado

(a) Solução provisória para o material desgastado

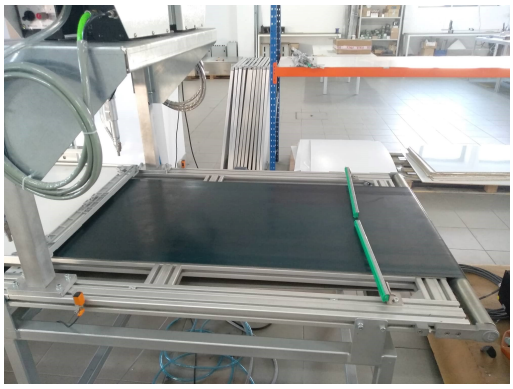


(b) Ajustar o binário da aparafusadora

Figura F.2: Testes Mesa fixação

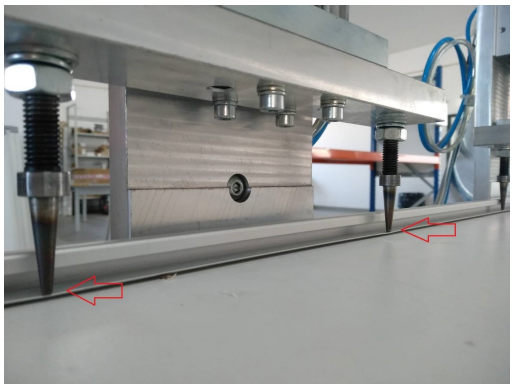


(a) Distância de 'C2'



(b) Base provisória para orientação

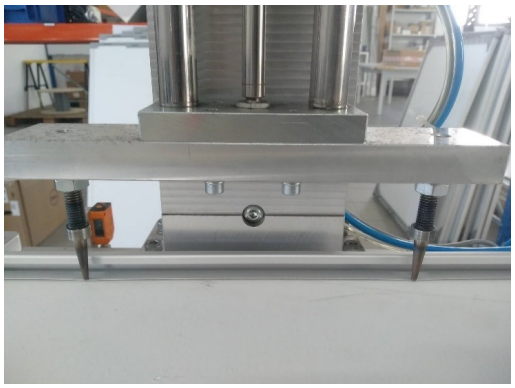
Figura F.3: Alinhamento do Quadro



(a) Pontas do cilindro na extremidade do metal

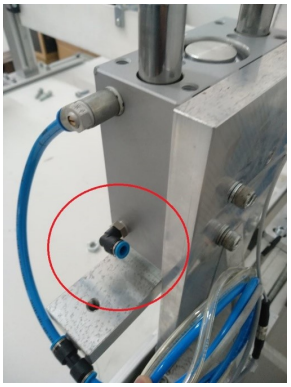
(b) Ponteiras com extremidades mais finas

(c) Resultado com uma única ponteira afiada

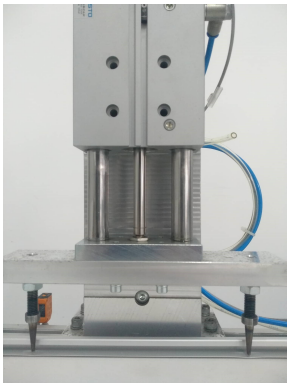


(d) Resultado com ponteiras iniciais à mesma altura

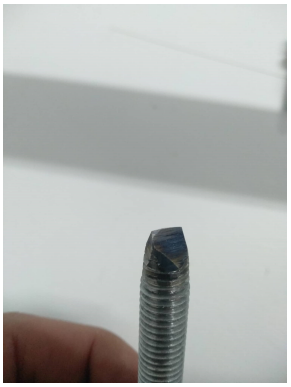
(e) Resultado com ponteiras afiadas à mesma altura



(f) Cilindro sem válvula de escape

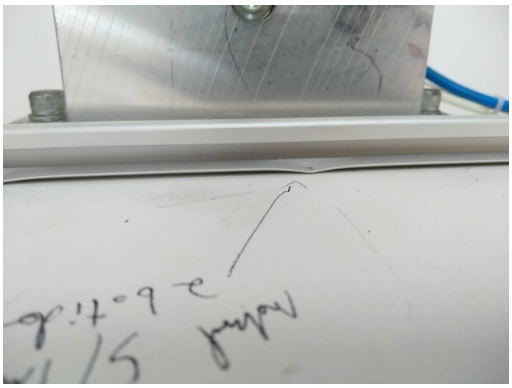


(g) Resultado sem válvula de escape

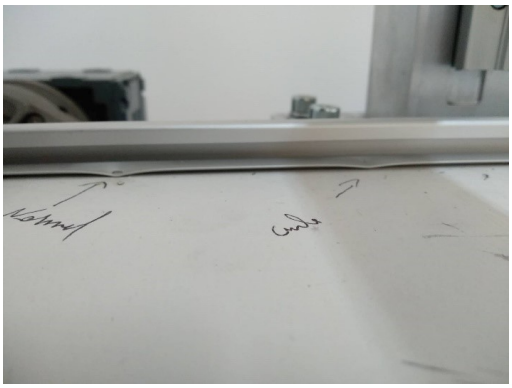


(h) Ponteira em forma de cunha

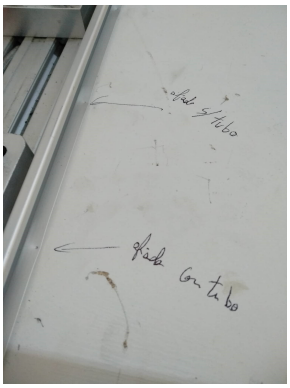
(i) Resultado sem válvula de escape



(j) Resultado com duas batidas



(k) Comparação entre ponteira normal e cunha (com válvula de escape)



(l) Comparação sem e com válcula de escape (Utilizado ponteira afiada)

Figura F.4: Testes Amolgar

Apêndice G

Sequência Final Da Mesa De Fixação

- Inicialização
    - Colocar tudo na posição inicial:
        - Vácuo desligado
        - Cilindros mesa recuados (menos cilindro do *stopper* 1)
        - Avançar *stopper* 1
- Quadro:
    - Dar acesso ao robô para ir apanhar o quadro completo
    - Esperar a confirmação do robô em como este colocou o quadro na posição final
- Deslocar quadro para a frente:
    - Sensor 1 ativo
    - Motor_1 ON
- Posição aparafusar:
    - Sensor_2 ativo (começa *timmer*)
    - Avançar cilindro base aparafusadora
    - Motor_1 OFF (*Timmer* termina)
- Aparafusar:
    - Avançar cilindro topo aparafusadora
    - Avançar cilindros aparafusadora
    - *Timer* (esperar que aparafuse)
- Recuar aparafusadora:
    - Recuar cilindros aparafusadora
    - Recuar cilindro topo aparafusadora
- Permitir deslocar quadro:
    - Recuar cilindros base aparafusadoras
    - Recuar *stopper* 1
- Deslocar quadro:

Motor_1 ON

Sensor_1 desativado(começa *timmer*)

Subir *stopper* 2

- Deslocar quadro para a trás:

  Motor_1 ON

- Posição aparafusar:

  Avançar cilindros base aparafusadoras

  Motor_1 OFF (*Timmer* termina)

- Aparafusar:

  Avançar cilindro topo aparafusadora

  Avançar cilindros aparafusadora

  Timer (esperar que aparafuse)

- Recuar aparafusadora:

  Recuar cilindros aparafusadora

  Recuar cilindro topo aparafusadora

- Amolgar quadro:

  Avançar cilindro *push* base

  Avançar cilindro *push* topo

- Permitir deslocar quadro:

  Recuar cilindros base aparafusadoras

  Recuar *stopper* 2

- Deslocar quadro para a frente:

  Motor_1 ON

Apêndice H

Programação Robô 2

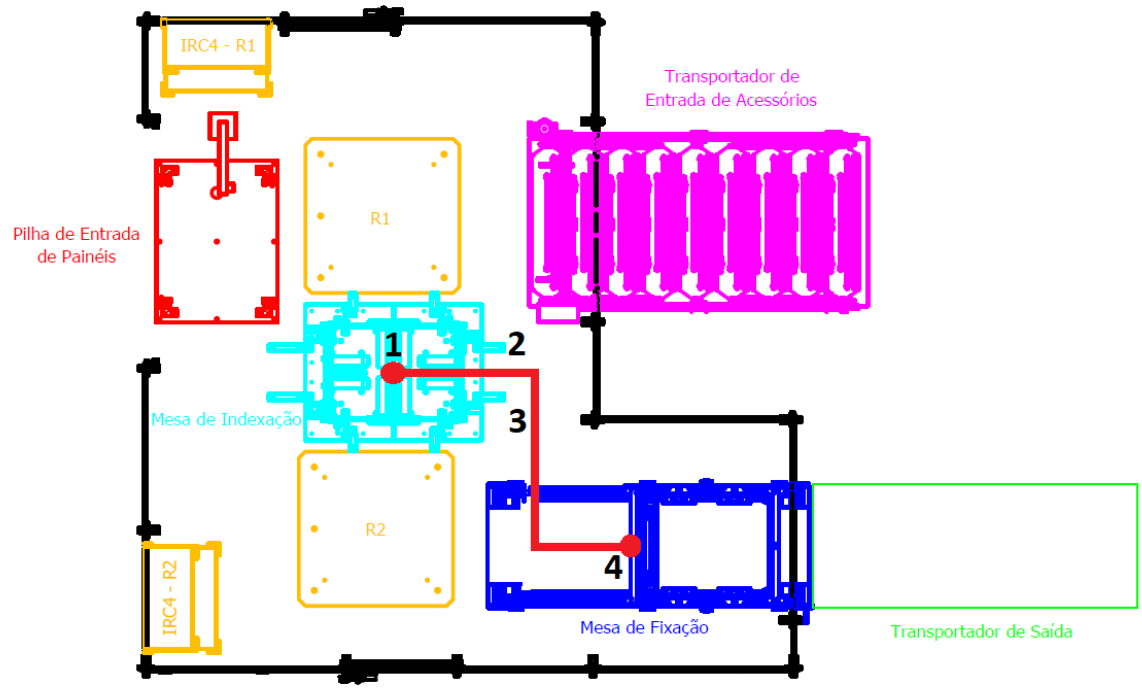


Figura H.1: Trajetória do Robô 2

Tabela H.1: Programas utilizados no robô 2

Programas_R2	
Cod. Programa	**Descrição**
1	Reserve R1
2	Pick Quadro Completo + Place Quadro Completo
3	Reserve
4	Reserve
5	Reserve

Tabela H.2: Áreas utilizadas pelo robô 2

Áreas_R2	
Num. Área	**Descrição**
1	Mesa montagem
2	Mesa aparafusar
3	Reserve
4	Reserve
5	Reserve

Tabela H.3: Programas finalizados pelo robô 2

Prog_Done_R2	
Num. trabalho realizado	**Descrição**
1	Prog Done 2
2	Reserve
3	Reserve
4	Reserve
5	Reserve

Tabela H.4: Áreas de colisão do robô 2

Anticollision_R2	
Num. Colisão	**Descrição**
1	Collision
2	Reserve
3	Reserve
4	Reserve
5	Reserve

Tabela H.5: Lista de erros do robô 2

Error_Code_R11	
Error	**Descrição**
120	Error Pick Painel
121	Error Lost Panel
122	Error Place Painel
123	Error Initialization
124	Reserve
125	Reserve

RobotSol KUKA Standard R2

18/02/2019

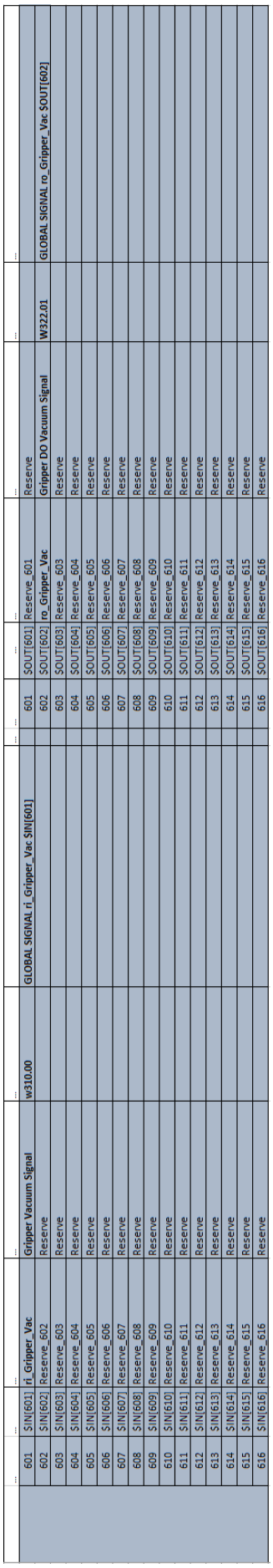
	512 ROBOT	INPUTS ROBOT/OUTPUTS PLC Symbol	Description	NOTES	Code to Work Visual	512 ROBOT	OUTPUTS ROBOT/ INPUTS PLC Symbol	Description	NOTES	Code to Work Visual
SYSTEM	1	$IN[1] $DRIVES_OFF	Drives Off (Inverse)	W300.00	GLOBAL SIGNAL $DRIVES_OFF $IN[1]	1	$OUT[1] $PERI_RDY	Drives Are Switched On	W311.00	GLOBAL SIGNAL $PERI_RDY $OUT[1]
	2	$IN[2] $EXT_START	Program Start	W300.01	GLOBAL SIGNAL $EXT_START $IN[2]	2	$OUT[2] $I_O_ACTCONF	Aut Ext Selected	W311.01	GLOBAL SIGNAL $I_O_ACTCONF $OUT[2]
	3	$IN[3] $DRIVES_ON	Drives ON	W300.02	GLOBAL SIGNAL $DRIVES_ON $IN[3]	3	$OUT[3] $STOPMESS	Error Message	W311.02	GLOBAL SIGNAL $STOPMESS $OUT[3]
	4	$IN[4] $MOVE_ENABLE	Move Enable	W300.03	GLOBAL SIGNAL $MOVE_ENABLE $IN[4]	4	$OUT[4] $PRO_ACT	Program Active	W311.03	GLOBAL SIGNAL $PRO_ACT $OUT[4]
	5	$IN[5] PGNO BIT1	Program Number Bit 1	W301.00	GLOBAL SIGNAL PGNO BIT1 $IN[5]	5	$OUT[5] $IN_HOME	Robot in Home Position	W311.04	GLOBAL SIGNAL $IN_HOME $OUT[5]
	6	$IN[6] PGNO BIT2	Program Number Bit 2	W301.01	GLOBAL SIGNAL PGNO BIT2 $IN[6]	6	$OUT[6] PGNO_REQ	Program Request	W311.05	GLOBAL SIGNAL PGNO_REQ $OUT[6]
	7	$IN[7] PGNO BIT3	Program Number Bit 3	W301.02	GLOBAL SIGNAL PGNO BIT3 $IN[7]	7	$OUT[7] $ON_PATH	Robot On Path	W311.06	GLOBAL SIGNAL $ON_PATH $OUT[7]
	8	$IN[8] PGNO BIT4	Program Number Bit 4	W301.03	GLOBAL SIGNAL PGNO BIT4 $IN[8]	8	$OUT[8] APPL_RUN	Aplication Run	W311.07	GLOBAL SIGNAL APPL_RUN $OUT[8]
	9	$IN[9] PGNO BIT5	Program Number Bit 5	W301.04	GLOBAL SIGNAL PGNO BIT5 $IN[9]	9	$OUT[9] $ROB_STOPPED	Robot Stopped	W312.00	GLOBAL SIGNAL $ROB_STOPPED $OUT[9]
	10	$IN[10] PGNO BIT6	Program Number Bit 6	W301.05	GLOBAL SIGNAL PGNO BIT6 $IN[10]	10	$OUT[10] $T1	Robot in T1 Mode	W312.01	GLOBAL SIGNAL $T1 $OUT[10]
	11	$IN[11] PGNO_PARITY	Parity Bit	W302.00	GLOBAL SIGNAL PGNO_PARITY $IN[11]	11	$OUT[11] $T2	Robot in T2 Mode	W312.02	GLOBAL SIGNAL $T2 $OUT[11]
	12	$IN[12] $CONF_MESS	Error Confirmation	W302.01	GLOBAL SIGNAL $CONF_MESS $IN[12]	12	$OUT[12] $AUT	Robot in Auto Mode	W312.03	GLOBAL SIGNAL $AUT $OUT[12]
	13	$IN[13] PGNO_VALID	Program Number Valid	W302.02	GLOBAL SIGNAL PGNO_VALID $IN[13]	13	$OUT[13] $EXT	Robot in Auto Ext Mode	W312.04	GLOBAL SIGNAL $EXT $OUT[13]
	14	$IN[14] Reserve_14	Reserve			14	$OUT[14] $USER_SAF	Operator safety closed	W312.05	GLOBAL SIGNAL $USER_SAF $OUT[14]
	15	$IN[15] Reserve_15	Reserve			15	$OUT[15] $ALARM_STOP	Alarm Stop	W312.06	GLOBAL SIGNAL $ALARM_STOP $OUT[15]
	16	$IN[16] Reserve_16	Reserve			16	$OUT[16] Reserve_16	Reserve		
	17	$IN[17] Reserve_17	Reserve			17	$OUT[17] Reserve_17	Reserve		
	18	$IN[18] Reserve_18	Reserve			18	$OUT[18] Reserve_18	Reserve		
	19	$IN[19] Reserve_19	Reserve			19	$OUT[19] Reserve_19	Reserve		
	20	$IN[20] Reserve_20	Reserve			20	$OUT[20] Reserve_20	Reserve		
	21	$IN[21] Reserve_21	Reserve			21	$OUT[21] Reserve_21	Reserve		
	22	$IN[22] Reserve_22	Reserve			22	$OUT[22] Reserve_22	Reserve		
	23	$IN[23] Reserve_23	Reserve			23	$OUT[23] Reserve_23	Reserve		
	24	$IN[24] Reserve_24	Reserve			24	$OUT[24] Reserve_24	Reserve		
GENERAL SIGNALS	25	$IN[25] di_Cell_Access_Req	Cell Access Request	W303.00	GLOBAL SIGNAL di_Cell_Access_Req $IN[25]	25	$OUT[25] do_Cell_Access_Granted	Cell Access Granted	W313.00	GLOBAL SIGNAL do_Cell_Access_Granted $OUT[25]
	26	$IN[26] Reserve_26	Reserve			26	$OUT[26] Reserve_26	Reserve		
	27	$IN[27] Reserve_27	Reserve			27	$OUT[27] Reserve_27	Reserve		
	28	$IN[28] Reserve_28	Reserve			28	$OUT[28] Reserve_28	Reserve		
	29	$IN[29] Reserve_29	Reserve			29	$OUT[29] Reserve_29	Reserve		
	30	$IN[30] Reserve_30	Reserve			30	$OUT[30] Reserve_30	Reserve		
	31	$IN[31] Reserve_31	Reserve			31	$OUT[31] Reserve_31	Reserve		
	32	$IN[32] Reserve_32	Reserve			32	$OUT[32] Reserve_32	Reserve		

ERRORS	33	SIN[33]	di_AckErrorCode_b1	Acknowledge Error Code Bit 1	W304.00	GLOBAL SIGNAL di_AckErrorCode_b1 SIN[33]		33	SOUT[33]	do_ErrorCode_b1	Error Code Bit 1	W314.00	GLOBAL SIGNAL do_ErrorCode_b1 $OUT[33]	
	34	SIN[34]	di_AckErrorCode_b2	Acknowledge Error Code Bit 2	W304.01	GLOBAL SIGNAL di_AckErrorCode_b2 SIN[34]		34	SOUT[34]	do_ErrorCode_b2	Error Code Bit 2	W314.01	GLOBAL SIGNAL do_ErrorCode_b2 $OUT[34]	
	35	SIN[35]	di_AckErrorCode_b3	Acknowledge Error Code Bit 3	W304.02	GLOBAL SIGNAL di_AckErrorCode_b3 SIN[35]		35	SOUT[35]	do_ErrorCode_b3	Error Code Bit 3	W314.02	GLOBAL SIGNAL do_ErrorCode_b3 $OUT[35]	
	36	SIN[36]	di_AckErrorCode_b4	Acknowledge Error Code Bit 4	W304.03	GLOBAL SIGNAL di_AckErrorCode_b4 SIN[36]		36	SOUT[36]	do_ErrorCode_b4	Error Code Bit 4	W314.03	GLOBAL SIGNAL do_ErrorCode_b4 $OUT[36]	
	37	SIN[37]	di_AckErrorCode_b5	Acknowledge Error Code Bit 5	W304.04	GLOBAL SIGNAL di_AckErrorCode_b5 SIN[37]		37	SOUT[37]	do_ErrorCode_b5	Error Code Bit 5	W314.04	GLOBAL SIGNAL do_ErrorCode_b5 $OUT[37]	
	38	SIN[38]	di_AckErrorCode_b6	Acknowledge Error Code Bit 6	W304.05	GLOBAL SIGNAL di_AckErrorCode_b6 SIN[38]		38	SOUT[38]	do_ErrorCode_b6	Error Code Bit 6	W314.05	GLOBAL SIGNAL do_ErrorCode_b6 $OUT[38]	
	39	SIN[39]	di_AckErrorCode_b7	Acknowledge Error Code Bit 7	W304.06	GLOBAL SIGNAL di_AckErrorCode_b7 SIN[39]		39	SOUT[39]	do_ErrorCode_b7	Error Code Bit 7	W314.06	GLOBAL SIGNAL do_ErrorCode_b7 $OUT[39]	
	40	SIN[40]	di_AckErrorCode_b8	Acknowledge Error Code Bit 8	W304.07	GLOBAL SIGNAL di_AckErrorCode_b8 SIN[40]		40	SOUT[40]	do_ErrorCode_b8	Error Code Bit 8	W314.07	GLOBAL SIGNAL do_ErrorCode_b8 $OUT[40]	
PROGRAMS	41	SIN[41]	di_ProgCode_b1	Program Code Bit 1	W305.00	GLOBAL SIGNAL di_ProgCode_b1 SIN[41]		41	SOUT[41]	do_ProgCodeReq_b1	Program Code Request Bit 1	W315.00	GLOBAL SIGNAL do_ProgCodeReq_b1 $OUT[41]	
	42	SIN[42]	di_ProgCode_b2	Program Code Bit 2	W305.01	GLOBAL SIGNAL di_ProgCode_b2 SIN[42]		42	SOUT[42]	do_ProgCodeReq_b2	Program Code Request Bit 2	W315.01	GLOBAL SIGNAL do_ProgCodeReq_b2 $OUT[42]	
	43	SIN[43]	di_ProgCode_b3	Program Code Bit 3	W305.02	GLOBAL SIGNAL di_ProgCode_b3 SIN[43]		43	SOUT[43]	do_ProgCodeReq_b3	Program Code Request Bit 3	W315.02	GLOBAL SIGNAL do_ProgCodeReq_b3 $OUT[43]	
	44	SIN[44]	di_ProgCode_b4	Program Code Bit 4	W305.03	GLOBAL SIGNAL di_ProgCode_b4 SIN[44]		44	SOUT[44]	do_ProgCodeReq_b4	Program Code Request Bit 4	W315.03	GLOBAL SIGNAL do_ProgCodeReq_b4 $OUT[44]	
	45	SIN[45]	di_ProgCode_b5	Program Code Bit 5	W305.04	GLOBAL SIGNAL di_ProgCode_b5 SIN[45]		45	SOUT[45]	do_ProgCodeReq_b5	Program Code Request Bit 5	W315.04	GLOBAL SIGNAL do_ProgCodeReq_b5 $OUT[45]	
	46	SIN[46]	di_ProgCode_b6	Program Code Bit 6	W305.05	GLOBAL SIGNAL di_ProgCode_b6 SIN[46]		46	SOUT[46]	do_ProgCodeReq_b6	Program Code Request Bit 6	W315.05	GLOBAL SIGNAL do_ProgCodeReq_b6 $OUT[46]	
	47	SIN[47]	di_ProgCode_b7	Program Code Bit 7	W305.06	GLOBAL SIGNAL di_ProgCode_b7 SIN[47]		47	SOUT[47]	do_ProgCodeReq_b7	Program Code Request Bit 7	W315.06	GLOBAL SIGNAL do_ProgCodeReq_b7 $OUT[47]	
	48	SIN[48]	di_ProgCode_b8	Program Code Bit 8	W305.07	GLOBAL SIGNAL di_ProgCode_b8 SIN[48]		48	SOUT[48]	do_ProgCodeReq_b8	Program Code Request Bit 8	W315.07	GLOBAL SIGNAL do_ProgCodeReq_b8 $OUT[48]	
	49	SIN[49]	Reserve					49	SOUT[49]	do_ProgDone_b1	Program Done Bit 1	W316.00	GLOBAL SIGNAL do_ProgDone_b1 $OUT[49]	
	50	SIN[50]	Reserve_50	Reserve				50	SOUT[50]	do_ProgDone_b2	Program Done Bit 2	W316.01	GLOBAL SIGNAL do_ProgDone_b2 $OUT[50]	
	51	SIN[51]	Reserve_51	Reserve				51	SOUT[51]	do_ProgDone_b3	Program Done Bit 3	W316.02	GLOBAL SIGNAL do_ProgDone_b3 $OUT[51]	
	52	SIN[52]	Reserve_52	Reserve				52	SOUT[52]	do_ProgDone_b4	Program Done Bit 4	W316.03	GLOBAL SIGNAL do_ProgDone_b4 $OUT[52]	
	53	SIN[53]	Reserve_53	Reserve				53	SOUT[53]	do_ProgDone_b5	Program Done Bit 5	W316.04	GLOBAL SIGNAL do_ProgDone_b5 $OUT[53]	
	54	SIN[54]	Reserve_54	Reserve				54	SOUT[54]	do_ProgDone_b6	Program Done Bit 6	W316.05	GLOBAL SIGNAL do_ProgDone_b6 $OUT[54]	
	55	SIN[55]	Reserve_55	Reserve				55	SOUT[55]	do_ProgDone_b7	Program Done Bit 7	W316.06	GLOBAL SIGNAL do_ProgDone_b7 $OUT[55]	
	56	SIN[56]	Reserve_56	Reserve				56	SOUT[56]	do_ProgDone_b8	Program Done Bit 8	W316.07	GLOBAL SIGNAL do_ProgDone_b8 $OUT[56]	
TASKS	57	SIN[57]	Reserve_57	Reserve				57	SOUT[57]	Reserve_57	Reserve			
	58	SIN[58]	Reserve_58	Reserve				58	SOUT[58]	Reserve_58	Reserve			
	59	SIN[59]	Reserve_59	Reserve				59	SOUT[59]	Reserve_59	Reserve			
	60	SIN[60]	Reserve_60	Reserve				60	SOUT[60]	Reserve_60	Reserve			
	61	SIN[61]	Reserve_61	Reserve				61	SOUT[61]	Reserve_61	Reserve			
	62	SIN[62]	Reserve_62	Reserve				62	SOUT[62]	Reserve_62	Reserve			
	63	SIN[63]	Reserve_63	Reserve				63	SOUT[63]	Reserve_63	Reserve			
	64	SIN[64]	Reserve_64	Reserve				64	SOUT[64]	Reserve_64	Reserve			
	65	SIN[65]	Reserve_65	Reserve				65	SOUT[65]	Reserve_65	Reserve			
	66	SIN[66]	Reserve_66	Reserve				66	SOUT[66]	Reserve_66	Reserve			
	67	SIN[67]	Reserve_67	Reserve				67	SOUT[67]	Reserve_67	Reserve			
	68	SIN[68]	Reserve_68	Reserve				68	SOUT[68]	Reserve_68	Reserve			
	69	SIN[69]	Reserve_69	Reserve				69	SOUT[69]	Reserve_69	Reserve			
	70	SIN[70]	Reserve_70	Reserve				70	SOUT[70]	Reserve_70	Reserve			
	71	SIN[71]	Reserve_71	Reserve				71	SOUT[71]	Reserve_71	Reserve			
	72	SIN[72]	Reserve_72	Reserve				72	SOUT[72]	Reserve_72	Reserve			

					W306.00	GLOBAL SIGNAL di_AreaReady_1 SIN[73]					W317.00	GLOBAL SIGNAL do_AreaReq_1 SOUT[73]	
	73	SIN[73]	di_AreaReady_1	Area Ready 1	W306.00	GLOBAL SIGNAL di_AreaReady_1 SIN[73]		73	SOUT[73]	do_AreaReq_1	Area Request 1	W317.00	GLOBAL SIGNAL do_AreaReq_1 SOUT[73]
	74	SIN[74]	di_AreaReady_2	Area Ready 2	W306.01	GLOBAL SIGNAL di_AreaReady_2 SIN[74]		74	SOUT[74]	do_AreaReq_2	Area Request 2	W317.01	GLOBAL SIGNAL do_AreaReq_2 SOUT[74]
	75	SIN[75]	Reserve_75	Reserve				75	SOUT[75]	Reserve_75	Reserve		
	76	SIN[76]	Reserve_76	Reserve				76	SOUT[76]	Reserve_76	Reserve		
	77	SIN[77]	Reserve_77	Reserve				77	SOUT[77]	Reserve_77	Reserve		
	78	SIN[78]	Reserve_78	Reserve				78	SOUT[78]	Reserve_78	Reserve		
	79	SIN[79]	Reserve_79	Reserve				79	SOUT[79]	Reserve_79	Reserve		
	80	SIN[80]	Reserve_80	Reserve				80	SOUT[80]	Reserve_80	Reserve		
	81	SIN[81]	Reserve_81	Reserve				81	SOUT[81]	Reserve_81	Reserve		
	82	SIN[82]	Reserve_82	Reserve				82	SOUT[82]	Reserve_82	Reserve		
	83	SIN[83]	Reserve_83	Reserve				83	SOUT[83]	Reserve_83	Reserve		
	84	SIN[84]	Reserve_84	Reserve				84	SOUT[84]	Reserve_84	Reserve		
	85	SIN[85]	Reserve_85	Reserve				85	SOUT[85]	Reserve_85	Reserve		
	86	SIN[86]	Reserve_86	Reserve				86	SOUT[86]	Reserve_86	Reserve		
	87	SIN[87]	Reserve_87	Reserve				87	SOUT[87]	Reserve_87	Reserve		
	88	SIN[88]	Reserve_88	Reserve				88	SOUT[88]	Reserve_88	Reserve		
	89	SIN[89]	Reserve_89	Reserve				89	SOUT[89]	Reserve_89	Reserve		
	90	SIN[90]	Reserve_90	Reserve				90	SOUT[90]	Reserve_90	Reserve		
	91	SIN[91]	Reserve_91	Reserve				91	SOUT[91]	Reserve_91	Reserve		
	92	SIN[92]	Reserve_92	Reserve				92	SOUT[92]	Reserve_92	Reserve		
	93	SIN[93]	Reserve_93	Reserve				93	SOUT[93]	Reserve_93	Reserve		
	94	SIN[94]	Reserve_94	Reserve				94	SOUT[94]	Reserve_94	Reserve		
	95	SIN[95]	Reserve_95	Reserve				95	SOUT[95]	Reserve_95	Reserve		
	96	SIN[96]	Reserve_96	Reserve				96	SOUT[96]	Reserve_96	Reserve		
	97	SIN[97]	Reserve_97	Reserve				97	SOUT[97]	Reserve_97	Reserve		
	98	SIN[98]	Reserve_98	Reserve				98	SOUT[98]	Reserve_98	Reserve		
	99	SIN[99]	Reserve_99	Reserve				99	SOUT[99]	Reserve_99	Reserve		
	100	SIN[100]	Reserve_100	Reserve				100	SOUT[100]	Reserve_100	Reserve		
	101	SIN[101]	Reserve_101	Reserve				101	SOUT[101]	Reserve_101	Reserve		
	102	SIN[102]	Reserve_102	Reserve				102	SOUT[102]	Reserve_102	Reserve		
	103	SIN[103]	Reserve_103	Reserve				103	SOUT[103]	Reserve_103	Reserve		
	104	SIN[104]	Reserve_104	Reserve				104	SOUT[104]	Reserve_104	Reserve		
WORK AREAS	105	SIN[105]	Reserve_105	Reserve				105	SOUT[105]	do_AreaRelease_1	Area Release 1	W318.00	GLOBAL SIGNAL do_AreaRelease_1 SOUT[105]
	106	SIN[106]	Reserve_106	Reserve				106	SOUT[106]	do_AreaRelease_2	Area Release 2	W318.01	GLOBAL SIGNAL do_AreaRelease_2 SOUT[106]
	107	SIN[107]	Reserve_107	Reserve				107	SOUT[107]	Reserve_107	Reserve		
	108	SIN[108]	Reserve_108	Reserve				108	SOUT[108]	Reserve_108	Reserve		
	109	SIN[109]	Reserve_109	Reserve				109	SOUT[109]	Reserve_109	Reserve		
	110	SIN[110]	Reserve_110	Reserve				110	SOUT[110]	Reserve_110	Reserve		
	111	SIN[111]	Reserve_111	Reserve				111	SOUT[111]	Reserve_111	Reserve		
	112	SIN[112]	Reserve_112	Reserve				112	SOUT[112]	Reserve_112	Reserve		
	113	SIN[113]	Reserve_113	Reserve				113	SOUT[113]	Reserve_113	Reserve		
	114	SIN[114]	Reserve_114	Reserve				114	SOUT[114]	Reserve_114	Reserve		
	115	SIN[115]	Reserve_115	Reserve				115	SOUT[115]	Reserve_115	Reserve		

PROCESSES	137	$IN[137]	di_ProcessDone_b1	Process Done 1	w307.00	GLOBAL SIGNAL di_ProcessDone_b1 $IN[137]		137	$OUT[137]	do_ProcessReq_b1	Process Request 1	W319.00	GLOBAL SIGNAL do_ProcessReq_b1 $OUT[137]			
	138	$IN[138]	di_ProcessDone_b2	Process Done 2	w307.01	GLOBAL SIGNAL di_ProcessDone_b2 $IN[138]		138	$OUT[138]	do_ProcessReq_b2	Process Request 2	W319.01	GLOBAL SIGNAL do_ProcessReq_b2 $OUT[138]			
	139	$IN[139]	di_ProcessDone_b3	Process Done 3	w307.02	GLOBAL SIGNAL di_ProcessDone_b3 $IN[139]		139	$OUT[139]	do_ProcessReq_b3	Process Request 3	W319.02	GLOBAL SIGNAL do_ProcessReq_b3 $OUT[139]			
	140	$IN[140]	di_ProcessDone_b4	Process Done 4	w307.03	GLOBAL SIGNAL di_ProcessDone_b4 $IN[140]		140	$OUT[140]	do_ProcessReq_b4	Process Request 4	W319.03	GLOBAL SIGNAL do_ProcessReq_b4 $OUT[140]			
	141	$IN[141]	di_ProcessDone_b5	Process Done 5	w307.04	GLOBAL SIGNAL di_ProcessDone_b5 $IN[141]		141	$OUT[141]	do_ProcessReq_b5	Process Request 5	W319.04	GLOBAL SIGNAL do_ProcessReq_b5 $OUT[141]			
	142	$IN[142]	di_ProcessDone_b6	Process Done 6	w307.05	GLOBAL SIGNAL di_ProcessDone_b6 $IN[142]		142	$OUT[142]	do_ProcessReq_b6	Process Request 6	W319.05	GLOBAL SIGNAL do_ProcessReq_b6 $OUT[142]			
	143	$IN[143]	di_ProcessDone_b7	Process Done 7	w307.06	GLOBAL SIGNAL di_ProcessDone_b7 $IN[143]		143	$OUT[143]	do_ProcessReq_b7	Process Request 7	W319.06	GLOBAL SIGNAL do_ProcessReq_b7 $OUT[143]			
	144	$IN[144]	di_ProcessDone_b8	Process Done 8	w307.07	GLOBAL SIGNAL di_ProcessDone_b8 $IN[144]		144	$OUT[144]	do_ProcessReq_b8	Process Request 8	W319.07	GLOBAL SIGNAL do_ProcessReq_b8 $OUT[144]			
	145	$IN[145]	Reserve_145	Reserve				145	$OUT[145]	Reserve_145	Reserve					
	146	$IN[146]	Reserve_146	Reserve				146	$OUT[146]	Reserve_146	Reserve					
	147	$IN[147]	Reserve_147	Reserve				147	$OUT[147]	Reserve_147	Reserve					
	148	$IN[148]	Reserve_148	Reserve				148	$OUT[148]	Reserve_148	Reserve					
	149	$IN[149]	Reserve_149	Reserve				149	$OUT[149]	Reserve_149	Reserve					
	150	$IN[150]	Reserve_150	Reserve				150	$OUT[150]	Reserve_150	Reserve					
	151	$IN[151]	Reserve_151	Reserve				151	$OUT[151]	Reserve_151	Reserve					
	152	$IN[152]	Reserve_152	Reserve				152	$OUT[152]	Reserve_152	Reserve					
	153	$IN[153]	Reserve_153	Reserve				153	$OUT[153]	Reserve_153	Reserve					
	154	$IN[154]	Reserve_154	Reserve				154	$OUT[154]	Reserve_154	Reserve					
	155	$IN[155]	Reserve_155	Reserve				155	$OUT[155]	Reserve_155	Reserve					
	156	$IN[156]	Reserve_156	Reserve				156	$OUT[156]	Reserve_156	Reserve					
	157	$IN[157]	Reserve_157	Reserve				157	$OUT[157]	Reserve_157	Reserve					
	158	$IN[158]	Reserve_158	Reserve				158	$OUT[158]	Reserve_158	Reserve					
	159	$IN[159]	Reserve_159	Reserve				159	$OUT[159]	Reserve_159	Reserve					
	160	$IN[160]	Reserve_160	Reserve				160	$OUT[160]	Reserve_160	Reserve					
	161	$IN[161]	Reserve_161	Reserve				161	$OUT[161]	Reserve_161	Reserve					
	162	$IN[162]	Reserve_162	Reserve				162	$OUT[162]	Reserve_162	Reserve					
	163	$IN[163]	Reserve_163	Reserve				163	$OUT[163]	Reserve_163	Reserve					
	164	$IN[164]	Reserve_164	Reserve				164	$OUT[164]	Reserve_164	Reserve					
	165	$IN[165]	Reserve_165	Reserve				165	$OUT[165]	Reserve_165	Reserve					
	166	$IN[166]	Reserve_166	Reserve				166	$OUT[166]	Reserve_166	Reserve					
	167	$IN[167]	Reserve_167	Reserve				167	$OUT[167]	Reserve_167	Reserve					
	168	$IN[168]	Reserve_168	Reserve				168	$OUT[168]	Reserve_168	Reserve					

GRIPPERS	169	$IN[169]	di_Gripper_Vac	Gripper DI Vacuum Signal	W308.00	GLOBAL SIGNAL di_Gripper_Vac $IN[169]		169	$OUT[169]	do_Gripper_Vac	Gripper DO Signal 1	W320.00	GLOBAL SIGNAL do_Gripper_Vac $OUT[169]	
	170	$IN[170]	Reserve_170	Reserve				170	$OUT[170]	Reserve_170	Reserve			
	171	$IN[171]	Reserve_171	Reserve				171	$OUT[171]	Reserve_171	Reserve			
	172	$IN[172]	Reserve_172	Reserve				172	$OUT[172]	Reserve_172	Reserve			
	173	$IN[173]	Reserve_173	Reserve				173	$OUT[173]	Reserve_173	Reserve			
	174	$IN[174]	Reserve_174	Reserve				174	$OUT[174]	Reserve_174	Reserve			
	175	$IN[175]	Reserve_175	Reserve				175	$OUT[175]	Reserve_175	Reserve			
	176	$IN[176]	Reserve_176	Reserve				176	$OUT[176]	Reserve_176	Reserve			
	177	$IN[177]	Reserve_177	Reserve				177	$OUT[177]	Reserve_177	Reserve			
	178	$IN[178]	Reserve_178	Reserve				178	$OUT[178]	Reserve_178	Reserve			
	179	$IN[179]	Reserve_179	Reserve				179	$OUT[179]	Reserve_179	Reserve			
	180	$IN[180]	Reserve_180	Reserve				180	$OUT[180]	Reserve_180	Reserve			
	181	$IN[181]	Reserve_181	Reserve				181	$OUT[181]	Reserve_181	Reserve			
	182	$IN[182]	Reserve_182	Reserve				182	$OUT[182]	Reserve_182	Reserve			
	183	$IN[183]	Reserve_183	Reserve				183	$OUT[183]	Reserve_183	Reserve			
	184	$IN[184]	Reserve_184	Reserve				184	$OUT[184]	Reserve_184	Reserve			
	185	$IN[185]	Reserve_185	Reserve				185	$OUT[185]	Reserve_185	Reserve			
	186	$IN[186]	Reserve_186	Reserve				186	$OUT[186]	Reserve_186	Reserve			
	187	$IN[187]	Reserve_187	Reserve				187	$OUT[187]	Reserve_187	Reserve			
	188	$IN[188]	Reserve_188	Reserve				188	$OUT[188]	Reserve_188	Reserve			
	189	$IN[189]	Reserve_189	Reserve				189	$OUT[189]	Reserve_189	Reserve			
	190	$IN[190]	Reserve_190	Reserve				190	$OUT[190]	Reserve_190	Reserve			
	191	$IN[191]	Reserve_191	Reserve				191	$OUT[191]	Reserve_191	Reserve			
	192	$IN[192]	Reserve_192	Reserve				192	$OUT[192]	Reserve_192	Reserve			
	193	$IN[193]	Reserve_193	Reserve				193	$OUT[193]	Reserve_193	Reserve			
	194	$IN[194]	Reserve_194	Reserve				194	$OUT[194]	Reserve_194	Reserve			
	195	$IN[195]	Reserve_195	Reserve				195	$OUT[195]	Reserve_195	Reserve			
	196	$IN[196]	Reserve_196	Reserve				196	$OUT[196]	Reserve_196	Reserve			
	197	$IN[197]	Reserve_197	Reserve				197	$OUT[197]	Reserve_197	Reserve			
	198	$IN[198]	Reserve_198	Reserve				198	$OUT[198]	Reserve_198	Reserve			
	199	$IN[199]	Reserve_199	Reserve				199	$OUT[199]	Reserve_199	Reserve			
	200	$IN[200]	Reserve_200	Reserve				200	$OUT[200]	Reserve_200	Reserve			
COLISIONS	201	$IN[201]	di_CollReady_1	Collision Ready 1	W309.00	GLOBAL SIGNAL di_CollReady_1 $IN[201]		201	$OUT[201]	do_CollRelease_1	Collision Release 1	W321.00	GLOBAL SIGNAL do_CollRelease_1 $OUT[201]	
	202	$IN[202]	Reserve_202	Reserve				202	$OUT[202]	Reserve_202	Reserve			
	203	$IN[203]	Reserve_203	Reserve				203	$OUT[203]	Reserve_203	Reserve			
	204	$IN[204]	Reserve_204	Reserve				204	$OUT[204]	Reserve_204	Reserve			
	205	$IN[205]	Reserve_205	Reserve				205	$OUT[205]	Reserve_205	Reserve			
	206	$IN[206]	Reserve_206	Reserve				206	$OUT[206]	Reserve_206	Reserve			
	207	$IN[207]	Reserve_207	Reserve				207	$OUT[207]	Reserve_207	Reserve			
	208	$IN[208]	Reserve_208	Reserve				208	$OUT[208]	Reserve_208	Reserve			



	601	$IN[601]	ri_Gripper_Vac	Gripper Vacuum Signal	w310.00	GLOBAL SIGNAL ri_Gripper_Vac $IN[601]		601	$OUT[601]	Reserve_601	Reserve		GLOBAL SIGNAL ro_Gripper_Vac $OUT[602]
	602	$IN[602]	Reserve_602	Reserve				602	$OUT[602]	ro_Gripper_Vac	Gripper DO Vacuum Signal	W322.01	
	603	$IN[603]	Reserve_603	Reserve				603	$OUT[603]	Reserve_603	Reserve		
	604	$IN[604]	Reserve_604	Reserve				604	$OUT[604]	Reserve_604	Reserve		
	605	$IN[605]	Reserve_605	Reserve				605	$OUT[605]	Reserve_605	Reserve		
	606	$IN[606]	Reserve_606	Reserve				606	$OUT[606]	Reserve_606	Reserve		
	607	$IN[607]	Reserve_607	Reserve				607	$OUT[607]	Reserve_607	Reserve		
	608	$IN[608]	Reserve_608	Reserve				608	$OUT[608]	Reserve_608	Reserve		
	609	$IN[609]	Reserve_609	Reserve				609	$OUT[609]	Reserve_609	Reserve		
	610	$IN[610]	Reserve_610	Reserve				610	$OUT[610]	Reserve_610	Reserve		
	611	$IN[611]	Reserve_611	Reserve				611	$OUT[611]	Reserve_611	Reserve		
	612	$IN[612]	Reserve_612	Reserve				612	$OUT[612]	Reserve_612	Reserve		
	613	$IN[613]	Reserve_613	Reserve				613	$OUT[613]	Reserve_613	Reserve		
	614	$IN[614]	Reserve_614	Reserve				614	$OUT[614]	Reserve_614	Reserve		
	615	$IN[615]	Reserve_615	Reserve				615	$OUT[615]	Reserve_615	Reserve		
	616	$IN[616]	Reserve_616	Reserve				616	$OUT[616]	Reserve_616	Reserve		

Figura H.2: Sinais *standard KUKA* utilizados entre robô e *PLC*

Apêndice I

Programação *HMI*

Figura I.1: Ecrã da janela de contactos

Figura I.2: Ecrã da janela dos sinais do robô 2

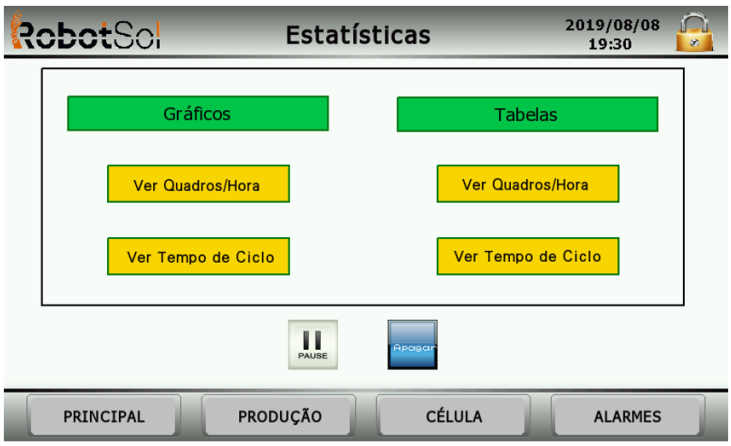


Figura I.3: Ecrã da janela de estatísticas

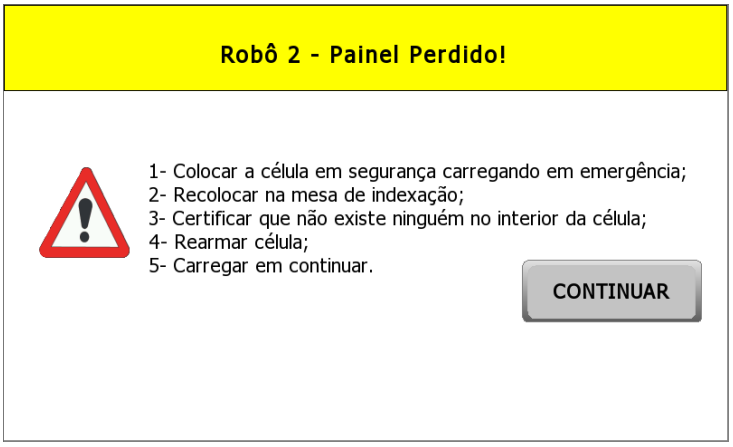


Figura I.4: Ecrã da janela do alarme de painel perdido

Apêndice J

Otimização Da Célula

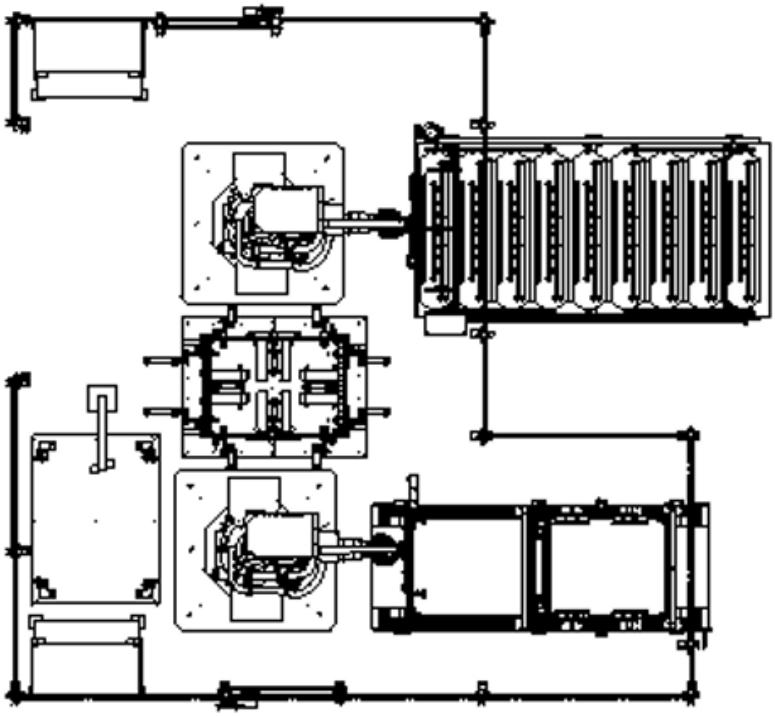
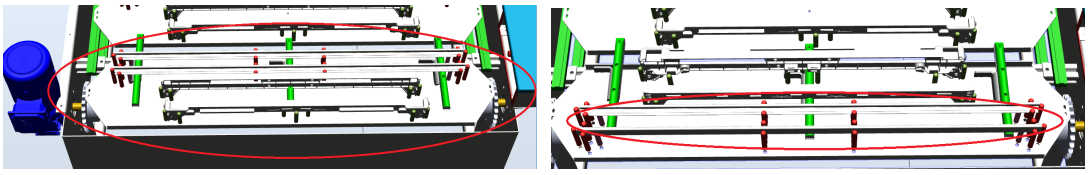


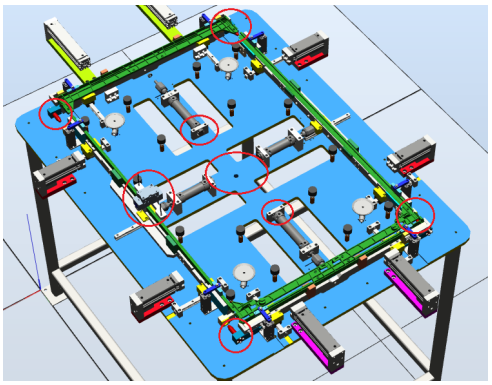
Figura J.1: *Layout* otimizado



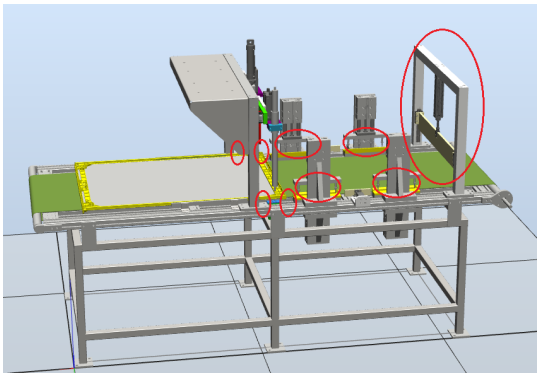
(a) Esquemático 3D transportador de entrada desatualizado

(b) Esquemático 3D transportador de entrada corrigido

Figura J.2: Esquemático 3D transportador de entrada



(a) Esquemático 3D mesa de indexação desatualizado



(b) Esquemático 3D mesa de fixação desatualizado

Figura J.3: Esquemático 3D mesas desatualizados

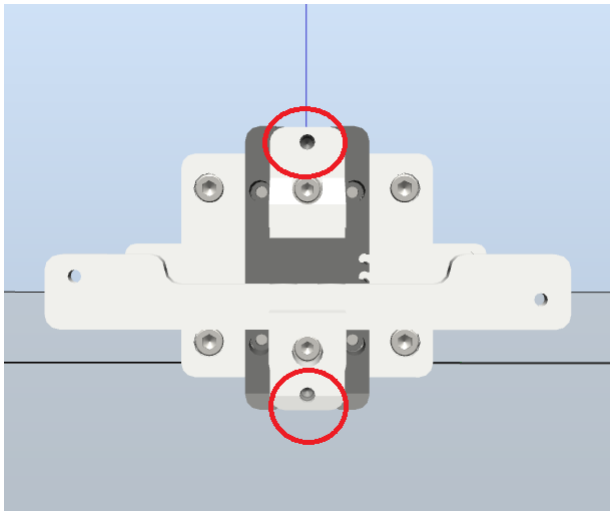
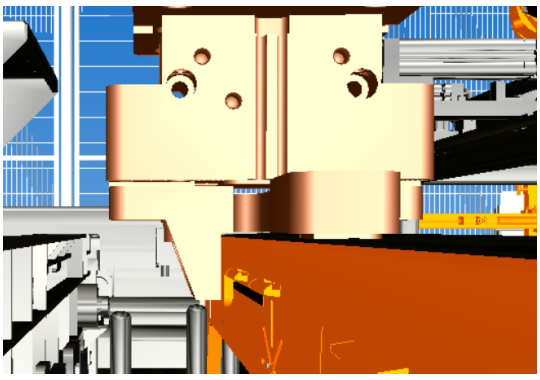
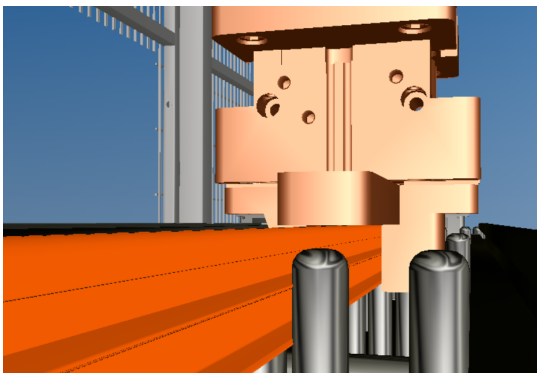


Figura J.4: Alteração do *gripper* do robô 1

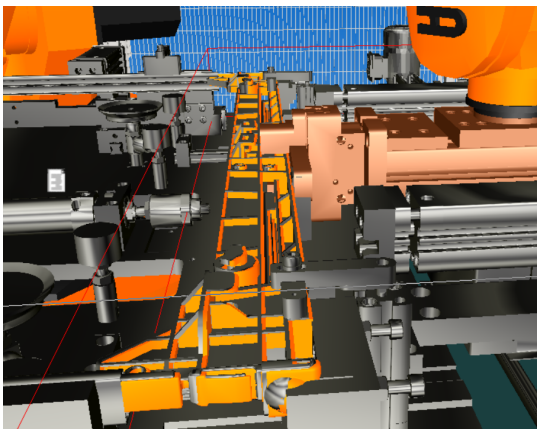


(a) Esquemático 3D *pick* peça plástica

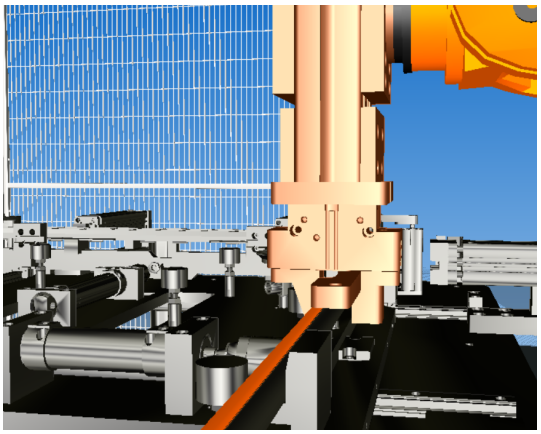


(b) Esquemático 3D *pick* peça metálica

Figura J.5: Esquemático 3D *pick* transportador de entrada

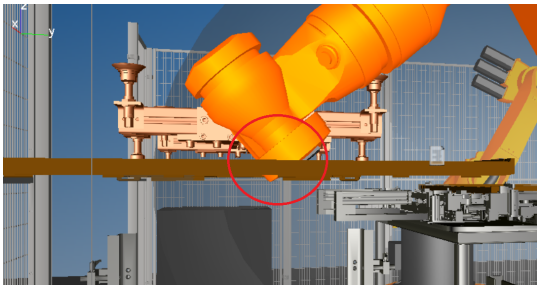


(a) Esquemático 3D *place* peça plástica

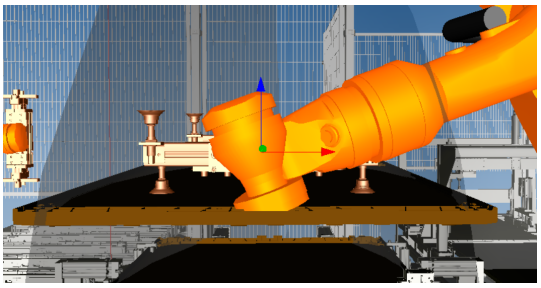


(b) Esquemático 3D *place* peça metálica

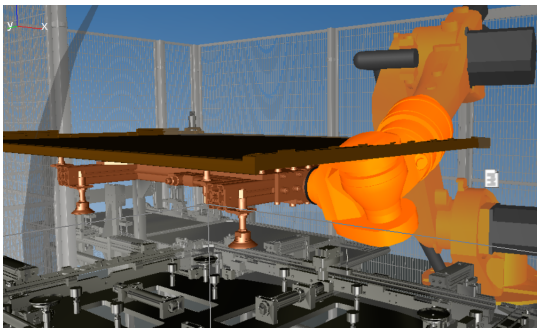
Figura J.6: Esquemático 3D *place* mesa de indexação



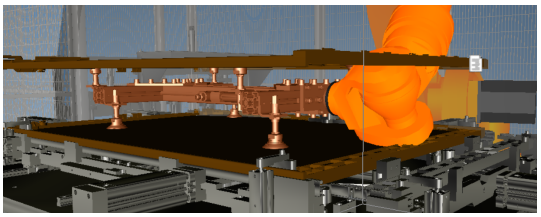
(a) Esquemático 3D ponto de aproximação do armazém de painéis



(b) Esquemático 3D ponto de aproximação da mesa de indexação

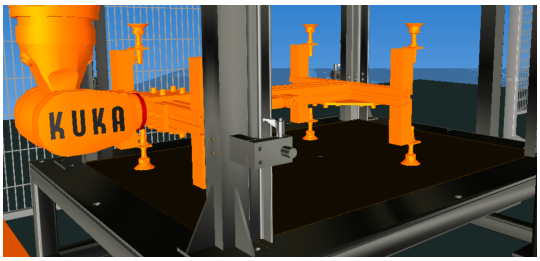


(c) Esquemático 3D ponto de aproximação do armazém de painéis

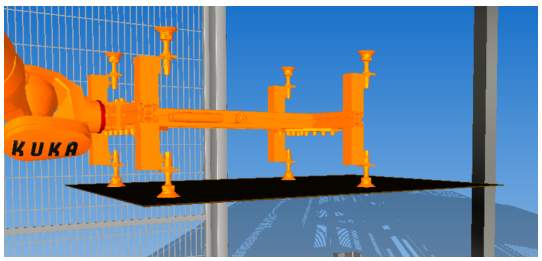


(d) [Esquemático 3D *pick* do quadro completo da mesa de indexação

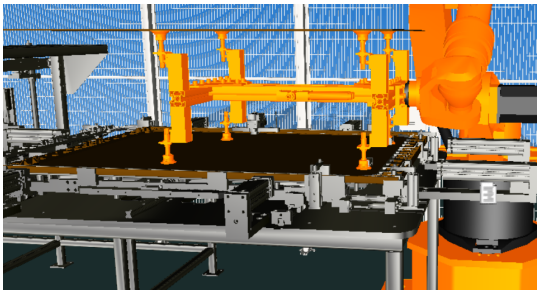
Figura J.7: Esquemático 3D da primeira simulação do robô 2



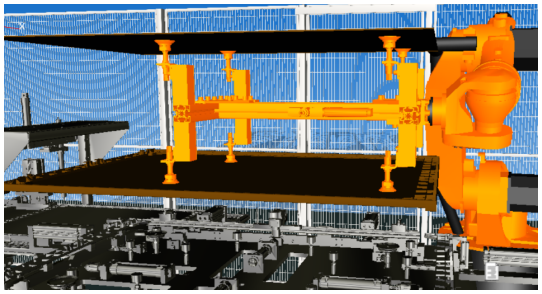
(a) Esquemático 3D ponto de *pick* do armazém de painéis



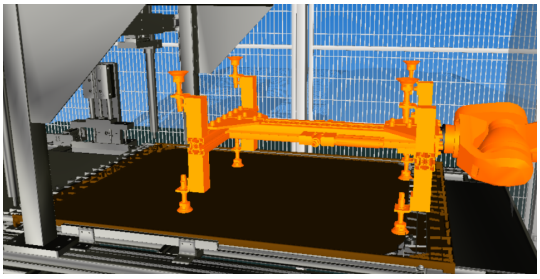
(b) Esquemático 3D ponto de aproximação do armazém de painéis



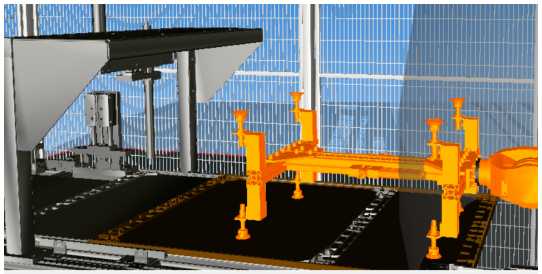
(c) Esquemático 3D ponto de *pick* da mesa de indexação



(d) Esquemático 3D ponto de aproximação da mesa de indexação



(e) Esquemático 3D ponto de *place* da mesa de fixação



(f) Esquemático 3D ponto de aproximação da mesa de fixação

Figura J.8: Esquemático 3D da simulação final do robô 2

Apêndice K

Testes Na Mesa De Fixação

Tabela K.1: Teste em série com as bases da aparafusadora recuada

Testes	Alinhamento	Furar - frente	Furar - atrás
1	Ok	Ok	Lado direito não aparafusou
2	Ok	Lado esquerdo não aparafusou	Ok
3	Ok	Lado esquerdo encravou (falta de binário)	Lado direito não aparafusou
4	Ok	Ok	Nenhum (parafusos ficaram em furos já existentes)
5	Ok	Ok	Nenhum aparafusou

Tabela K.2: Teste em série com as bases da aparafusadora avançadas

Testes	Alinhamento	Furar - frente	Furar - atrás
1	Ok	Lado direito não aparafusou	Ok
2	Ok	Parcialmente Ok (não entrou totalmente de um lado)	Nenhum aparafusou
3	Ok	Ok	Ok
4	Ok	Ok	Ok
5	Ok	Ok	Ok

Tabela K.3: Binário das aparafusadoras no nível 4 (13 voltas)

Testes	Lado Direito	Lado Esquerdo
1	Não furou	Ok
2	Ok	Ok
3	Ok	Ok
4	Não furou (furou pouco)	Ok
5	Ok	Ok

Tabela K.4: Binário da aparafusadora direita com 15 voltas

Testes	Lado Direito	Lado Esquerdo
1	Ok	Não furou
2	Ok	Ok
3	Ok	Ok
4	Ok	Ok
5	Ok	Encravou ao furar

Tabela K.5: Binário da aparafusadora esquerda com 15 voltas

Testes	Lado Direito	Lado Esquerdo
1	Ok	Não furou
2	Ok	Não furou
3	Ok	Não furou
4	Ok	Não furou
5	Ok	Ok

Tabela K.6: Binário da aparafusadora esquerda com 20 voltas

Testes	Lado Direito	Lado Esquerdo
1	Ok	Ok
2	Ok	Ok
3	Ok	Ok
4	Ok	Ok
5	Ok	Ok